

SMC-TR-92-52

AD-A258 872



(2)

AEROSPACE REPORT NO.  
TR-92(2930)-4

Plotting the Magnitude and Direction  
of Waveguide Surface Currents  
in Three Dimensions

Prepared by

T. J. PETERS  
Communications Systems Subdivision

DTIC  
ELECTED  
DEC 8 1992  
S C D

30 September 1992

Prepared for

SPACE AND MISSILE SYSTEMS CENTER  
AIR FORCE MATERIEL COMMAND  
Los Angeles Air Force Base  
P.O. Box 92960  
Los Angeles, CA 90009-2960

Engineering and Technology Group

THE AEROSPACE CORPORATION  
El Segundo, California

APPROVED FOR PUBLIC RELEASE;  
DISTRIBUTION UNLIMITED

92-30955



1228

This report was submitted by The Aerospace Corporation, El Segundo, CA 90245-4691, under Contract No. F04701-88-C-0089 with the Space Systems Division, P. O. Box 92960, Los Angeles, CA 90009-2960. It was reviewed and approved for The Aerospace Corporation by J. M. Straus, Principal Director, Communications Systems Subdivision. James Goble, Capt, USAF, was the project officer for the Mission-Oriented Investigation and Experimentation (MOIE) program.

This report has been reviewed by the Public Affairs Office (PAS) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication. Publication of this report does not constitute Air Force approval of the report's findings or conclusions. It is published only for the exchange and stimulation of ideas.

272  
QUANG BUI, Lt, USAF  
MOIE Program Manager

James R. Goble  
James R. Goble, Capt, USAF  
DSCS Advanced Technology Manager

## UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

## REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified		1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE			
4. PERFORMING ORGANIZATION REPORT NUMBER(S) TR-92(2930)-4		5. MONITORING ORGANIZATION REPORT NUMBER(S) SMC-TR-92-52	
6a. NAME OF PERFORMING ORGANIZATION The Aerospace Corporation Communications Systems Subdivision	6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Space and Missile Systems Center	
6c. ADDRESS (City, State, and ZIP Code) El Segundo, CA 90245		7b. ADDRESS (City, State, and ZIP Code) Los Angeles Air Force Base Los Angeles, CA 90009-2960	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION	8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F04701-88-C-0089	
8c. ADDRESS (City, State, and ZIP Code)		10. SOURCE OF FUNDING NUMBERS	
		PROGRAM ELEMENT NO.	PROJECT NO.
		TASK NO.	WORK UNIT ACCESSION NO.
11. TITLE (Including Security Classification) Plotting the Magnitude and Direction of Waveguide Surface Currents in Three Dimensions			
12. PERSONAL AUTHOR(S) Peters, T. J.			
13a. TYPE OF REPORT	13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Year, Month, Day) 30 September 1992	15. PAGE COUNT 119
16. SUPPLEMENTARY NOTATION			
17. COSATI CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number) Computer graphics Electromagnetic waveguides	
18. ABSTRACT (Continue on reverse if necessary and identify by block number)  A method is presented to plot the magnitude and direction of the current induced on the inside surface of a rectangular or circular waveguide. The method requires the geometry of each waveguide to be modeled using a triangular mesh and assumes that the surface current values are known only at the vertices. Since a simple mapping exists between a two-dimensional rectangular region and the surface of a rectangular or circular waveguide, the plotting algorithm is designed to process two-dimensional data and then perform a mapping to a three-dimensional surface.			
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT. <input type="checkbox"/> DTIC USERS		21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL		22b. TELEPHONE (Include Area Code)	22c. OFFICE SYMBOL

## Contents

I.	Introduction .....	5
II.	Drawing Vectors on Triangular Grids .....	7
III.	Initial, Terminal, and Critical Points .....	11
IV.	TE Wave Surface Currents.....	13
V.	Conclusion .....	19
	References .....	21
	Appendix A: Vector-Line Differential Equation Solution ...	23
	Appendix A References .....	27
	Appendix B: Computer Programs.....	29
	Appendix B References .....	117
	About the Author .....	119

DTIC QUALITY INSPECTED 8

Accession For	
NTIS GRADE	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unsolicited	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Assist and/or	
Dist	Special
A-1	

## Figures

1. Node and Triangle Connections .....	8
2. Critical Points for Diverging, Converging, or Closed-Loop Vector Fields .....	12
3. Magnitude and Direction of TE <sub>10</sub> Surface Current Plotted on Unfolded Rectangular Waveguide .....	15
4. Magnitude and Direction of TE <sub>10</sub> Surface Current Plotted on Rectangular Waveguide .....	16
5. Magnitude and Direction of TE <sub>11</sub> Surface Current Plotted on Unfolded Circular Waveguide .....	17
6. Magnitude and Direction of TE <sub>11</sub> Surface Current Plotted on Circular Waveguide .....	18

## I. INTRODUCTION

Understanding the flow of energy through a bounded waveguide region is enhanced by the ability to visualize surface-current vector fields. Also, determining the position and size of slots cut into the wall of a waveguide requires a knowledge of the current-flow magnitude and direction on the inside surface. Applications have been developed to represent waveguide modes [1]-[3], radiation fields [4], and static fields [5]-[6]; however, all are limited to only two dimensions. These methods usually exclude the flow direction of each vector and show only the tangent line. The magnitude either is not shown or is represented simply by variable line density or line thickness.

The purpose of this study is to develop a method of plotting the magnitude and direction of numerically computed current induced on the inside surface of a rectangular or circular waveguide. The inside surface is assumed to be described by a triangular mesh. This is consistent with a finite-element analysis using tetrahedral elements of the interior of a waveguide. Since the waveguide surface can be described by two variables of an orthogonal coordinate system, the vector field can be represented by a simple two-dimensional to three-dimensional mapping. This allows all processing to be done in two dimensions. The procedure will be to combine the three-dimensional contoured-surface representation technique developed in [7]-[8] with a two-dimensional vector drawing algorithm. The flow will be indicated by a continuous line with arrowhead markers, and the normalized magnitude by grayscale-shaded contours.

## II. DRAWING VECTORS ON TRIANGULAR GRIDS

Tetrahedral volume elements are used frequently for the finite-element analysis of the interior of a waveguide. This is especially true if the waveguide is nonuniformly filled, contains a scattering obstacle, or has a nonplanar surface. The tetrahedral element has four sides, which are all triangles. This implies that the surface of the waveguide is approximated by a triangular mesh, so that the numerical value of both vector components is computed at each node. Since three points determine a plane, a linear interpolation function is implied over every triangle. Therefore, each vector component has piecewise continuity between adjacent triangles.

The numbering scheme for the nodes and triangles of a region is completely arbitrary. Figure 1 indicates a possible configuration for six nodes and four triangles. The convention used in this study, to establish the connection between a triangle number and the three associated node numbers, is to store the relationship in a matrix called the triangle-to-node connection matrix, denoted by **C**. This matrix is defined when the original triangular grid is formed. The relevant matrix for the configuration of Fig. 1 is given as

$$\mathbf{C} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix} \quad (1)$$

where each row represents a triangle and each column represents one of the three triangle nodes. This matrix is used for the indirect addressing of the coordinates of each triangle.

The matrix **C** implicitly contains enough information to yield the triangle-to-triangle connection relationship between a particular triangle's three vertex nodes and all other triangles connected to one of these nodes. Unfortunately, to establish this relationship the entire matrix must be searched. This is quite time consuming, but it can be avoided by precomputing an auxiliary matrix called the node-to-triangle connection matrix, denoted by **T**. For the geometry of Fig. 1, **T** is given by

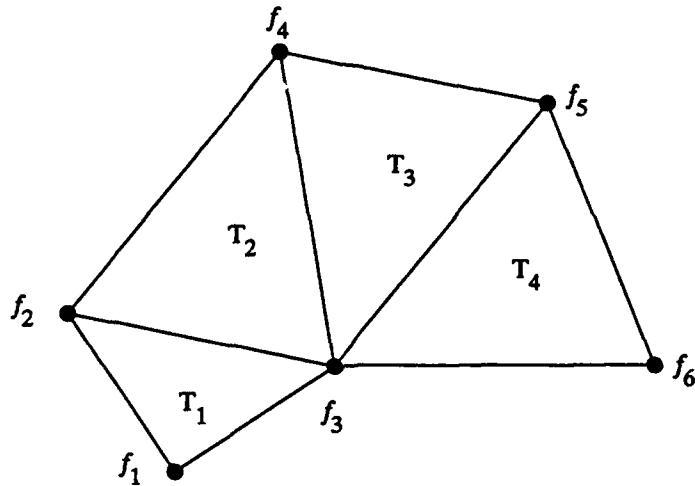


Fig. 1. Node and Triangle Connections

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 & 0 & 0 \\ 1 & 2 & 3 & 4 & 0 & 0 \\ 2 & 3 & 0 & 0 & 0 & 0 \\ 3 & 4 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (2)$$

where each row represents a node number and each column gives a triangle number connected to that node. The formation of this matrix is trivial once  $\mathbf{C}$  is known. For a triangular grid, there will be no more than six triangles connected to each node. This implies that there are no more than twelve triangles connected to any particular triangle

The interpolation function of a vector in the plane of each triangle is given by

$$\bar{f} = f_x \hat{x} + f_y \hat{y} \quad (3)$$

where

$$f_y = Ax + By + E \quad (4)$$

$$f_x = Cx + Dy + F. \quad (5)$$

By equating the function to the numerical values at the three vertices of the triangle,

the coefficients are computed as

$$A = [f_{y1}(y_2 - y_3) - f_{y2}(y_1 - y_3) + f_{y3}(y_1 - y_2)]/w \quad (6)$$

$$B = [-f_{y1}(x_2 - x_3) + f_{y2}(x_1 - x_3) - f_{y3}(x_1 - x_2)]/w \quad (7)$$

$$E = [f_{y1}(x_2y_3 - x_3y_2) - f_{y2}(x_1y_3 - x_3y_1) + f_{y3}(x_1y_2 - x_2y_1)]/w \quad (8)$$

$$C = [f_{x1}(y_2 - y_3) - f_{x2}(y_1 - y_3) + f_{x3}(y_1 - y_2)]/w \quad (9)$$

$$D = [-f_{x1}(x_2 - x_3) + f_{x2}(x_1 - x_3) - f_{x3}(x_1 - x_2)]/w \quad (10)$$

$$F = [f_{x1}(x_2y_3 - x_3y_2) - f_{x2}(x_1y_3 - x_3y_1) + f_{x3}(x_1y_2 - x_2y_1)]/w \quad (11)$$

where

$$w = x_1y_2 - x_1y_3 - x_2y_1 + x_3y_1 + x_2y_3 - x_3y_2. \quad (12)$$

For a given an observation point, the appropriate triangle is located using C and T, then the magnitude and direction of the vector field can be computed using Eqs. (4) and (5).

Although each component of the interpolated vector field changes linearly across each triangle, the path of the vector itself is nonlinear. The slope of a line tangent to the vector field at any point in the triangle is given by

$$\frac{dy}{dx} = \frac{Ax + By + E}{Cx + Dy + F}. \quad (13)$$

This is a nonlinear first-order differential equation. For a given starting point within a triangle, a vector path is uniquely defined by the solution to this differential equation. (The solution is given in Appendix A.) Unfortunately, the solution is an implicit function of the form  $W(x, y) = 0$ , which is not very convenient for rapid plotting. Fortunately, the method proposed and used in this study is much faster and easier to implement. Given some initial point, the vector is simply incremented in the positive or negative direction. This simple method allows arrows to be placed and accurately spaced on the lines. Also, the plotting speed is controlled entirely by the step size. The step size could be adaptively adjusted to reflect the variation in the vector direction, although that has not been attempted here. The vector position is

incremented as

$$x^{n+1} = x^n + \Delta \frac{f_x^n}{|\tilde{f}^n|} \quad (14)$$

$$y^{n+1} = y^n + \Delta \frac{f_y^n}{|\tilde{f}^n|} \quad (15)$$

where  $\Delta$  is a positive or negative adaptive step size and

$$|\tilde{f}^n| = \sqrt{(f_x^n)^2 + (f_y^n)^2}. \quad (16)$$

The vector length is accumulated as

$$l^{n+1} = l^n + \sqrt{(x^{n+1} - x^n)^2 + (y^{n+1} - y^n)^2} \quad (17)$$

and is used to provide equal spacing between arrowhead markers.

This method is designed for two-dimensional vector fields. However, since there is a point-to-point mapping between a rectangular two-dimensional region and the corresponding surface of the three-dimensional waveguide, then the vector lines can be mapped onto the waveguide surface. The three-dimensional surface is drawn by the method developed in [7]-[8]. Once this is complete, the visible portion of each vector line is drawn onto the surface.

### III. INITIAL, TERMINAL, AND CRITICAL POINTS

Each vector has an initial point and a terminal point. In some cases these points may even coincide. Choosing an appropriate initial point requires some knowledge about the points at which the vector magnitude goes to zero. These are called critical points [9]-[12]. For an interactive computer application, a convenient way to locate the critical points is to view a contour plot of the magnitude of one or both of the field components. A region can be selected by means of a cursor and a limited numerical search can be performed. Usually, a vector's initial point is in a region of high magnitude and a terminal point is either a boundary point or a critical point.

Several types of critical points can be used to characterize the flow of a vector field. Each type is based on the eigenvalues of the Jacobian matrix evaluated in the vicinity of the critical point. Let  $\mathbf{J}$  denote the Jacobian of the vector components at some specified point so that

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_x}{\partial x} & \frac{\partial f_x}{\partial y} \\ \frac{\partial f_y}{\partial x} & \frac{\partial f_y}{\partial y} \end{bmatrix} = \begin{bmatrix} C & D \\ A & B \end{bmatrix}. \quad (18)$$

For a triangular grid with vertex-specified vector values, the Jacobian is constant over each triangle. The type of critical point can be determined by looking at the eigenvalues of the Jacobian for a particular triangle. Letting  $\mathbf{I}$  denote the identity matrix, the eigenvalue equation is formed by taking the determinant

$$\det|\mathbf{J} - \gamma\mathbf{I}| = 0 \quad (19)$$

which yields a quadratic equation in the variable  $\gamma$ :

$$\gamma^2 - (B + C)\gamma + BC - AD = 0. \quad (20)$$

This equation has the solution

$$\gamma = \gamma_r + j\gamma_i = \frac{1}{2} \left[ (B + C) \pm \sqrt{(B + C)^2 - 4(BC - AD)} \right]. \quad (21)$$

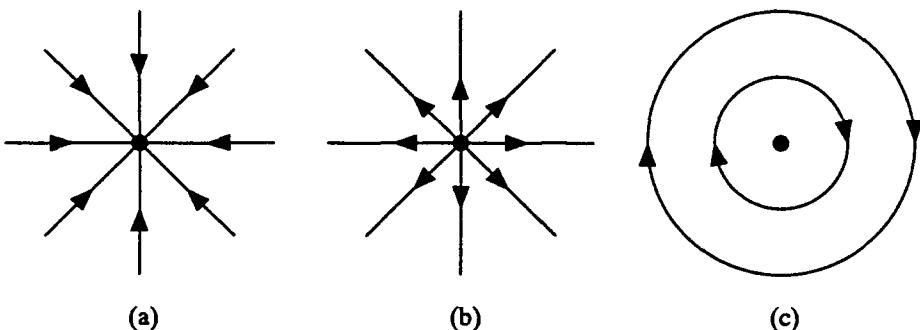


Fig. 2. Critical Points for (a) Diverging, (b) Converging, or (c) Closed Loop Vector Fields

Figure 2 shows the most common critical points encountered in plotting vector fields in electromagnetics. Figure 2(a) shows a field that converges to a point; this is represented mathematically as  $\gamma_r < 0$  and  $\gamma_i = 0$ . Figure 2(b) shows a field that diverges from a point; this is represented mathematically as  $\gamma_r > 0$  and  $\gamma_i = 0$ . Figure 2(c) shows a field that forms closed loops; this is represented mathematically as  $\gamma_r = 0$  and  $\gamma_i \neq 0$ . The diverging or converging critical points are sufficient to describe a surface-current vector field. Closed-loop critical points are necessary for drawing propagating electric and magnetic fields, such as those in the interior of the waveguide.

Once the critical points have been established, appropriate initial points can be specified. Since the vector-direction error accumulates with each increment, vectors are started at points of large magnitude so that roundoff error can be minimized. Each vector is drawn in two directions until a terminal point is reached. This reduces the error by half and causes the greatest error to occur at the terminal points (which are usually critical points). It can also help provide symmetry in the vector lines when appropriate. For the purposes of this study, vectors are drawn in groups that have initial points spaced equally along a line. The boundary points for each line can be easily controlled interactively by a cursor.

#### IV. TE WAVE SURFACE CURRENTS

Although the plotting procedure is designed to handle numerically computed data, for convenience it is tested by means of well-known analytic expressions for the surface current induced on the inside of a rectangular or circular waveguide excited by transverse electric (TE) waves. The magnitude and direction of the current flow is shown by superimposing the vector lines onto a contour plot of the normalized magnitude. The vector field can be viewed either on a two-dimensional surface or on the actual three-dimensional surface.

The  $\text{TE}_{mn}$ -mode magnetic-field components for a rectangular waveguide centered about the origin such that  $-a/2 \leq x \leq a/2$  and  $-b/2 \leq y \leq b/2$  are given by

$$H_x = -\frac{m\pi\beta H_0}{ak_c^2} \sin \psi_x \cos \psi_y \sin(\omega t - \beta z) \quad (22)$$

$$H_y = -\frac{n\pi\beta H_0}{bk_c^2} \cos \psi_x \sin \psi_y \sin(\omega t - \beta z) \quad (23)$$

$$H_z = H_0 \cos \psi_x \cos \psi_y \cos(\omega t - \beta z) \quad (24)$$

where  $\beta = \sqrt{k_0^2 - k_c^2}$ ,  $k_x = m\pi/a$ ,  $k_y = n\pi/b$ ,  $k_c^2 = k_x^2 + k_y^2$ ,  $\psi_x = k_x(x - a/2)$ , and  $\psi_y = k_y(y - b/2)$ . Defining a unit vector normal to each inside wall as  $\hat{n}$ , the surface current components on the inside of the waveguide are given by

$$K_x = n_y H_z - n_z H_y \quad (25)$$

$$K_y = n_z H_x - n_x H_z \quad (26)$$

$$K_z = n_x H_y - n_y H_x \quad (27)$$

Figure 3 shows the magnitude and direction of the surface-current for the  $\text{TE}_{10}$ -mode for  $t = 0$ . The waveguide dimensions are  $a = 1\lambda_0$ ,  $b = 0.5\lambda_0$ , and  $0 \leq z \leq 5\pi/\beta$ . There are 27 nodes across each wide side, 9 samples across each narrow side, and 62 samples along the longitudinal direction. This results in a total of 4278 nodes and 8296 triangles. Figure 4 shows the same vector field mapped to the outside surface of the waveguide.

The  $\text{TE}_{nm}$ -mode magnetic field components for a circular waveguide of radius  $a$

are given by

$$H_\phi = \frac{-nk_0^2 H_0}{k_c \beta} \frac{J_n(k_c r)}{k_c r} \sin(n\phi) \sin(\omega t - \beta z) \quad (28)$$

$$H_z = H_0 J_n(k_c r) \cos(n\phi) \cos(\omega t - \beta z) \quad (29)$$

where  $k_c = p'_{nm}/a$ . The variable  $p'_{nm}$  is the  $m$ th root of  $dJ_n(k_c r)/dr = 0$  evaluated at  $r = a$ . The surface current components on the inside of the waveguide are given by

$$K_\phi = H_z \quad (21)$$

$$K_z = -H_\phi. \quad (22)$$

Figure 5 shows the magnitude and direction of the surface current for the TE<sub>11</sub> mode for  $t = 0$ . The waveguide has radius  $a = 1\lambda_0$  and length  $5\pi/\beta$  in the range  $0 \leq z \leq 5\pi/\beta$ . There are 45 nodes in the  $\phi$  direction and 62 nodes in the longitudinal direction, for a total of 2790 nodes and 5368 triangles. Figure 6 shows the same vector field mapped to the outside surface of the waveguide.

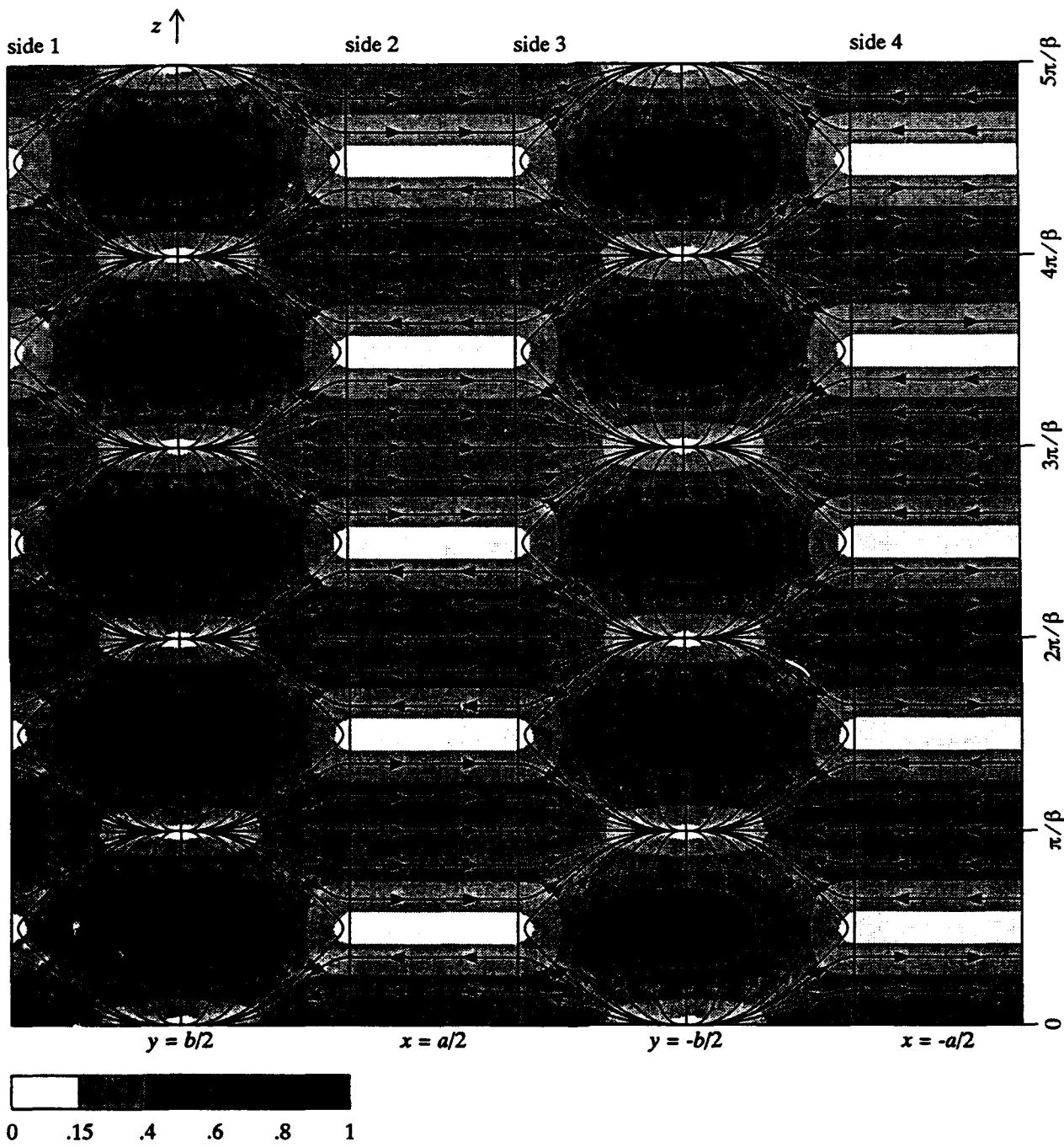


Fig. 3. Magnitude and Direction of  $TE_{10}$  Surface Current Plotted on Unfolded Rectangular Waveguide

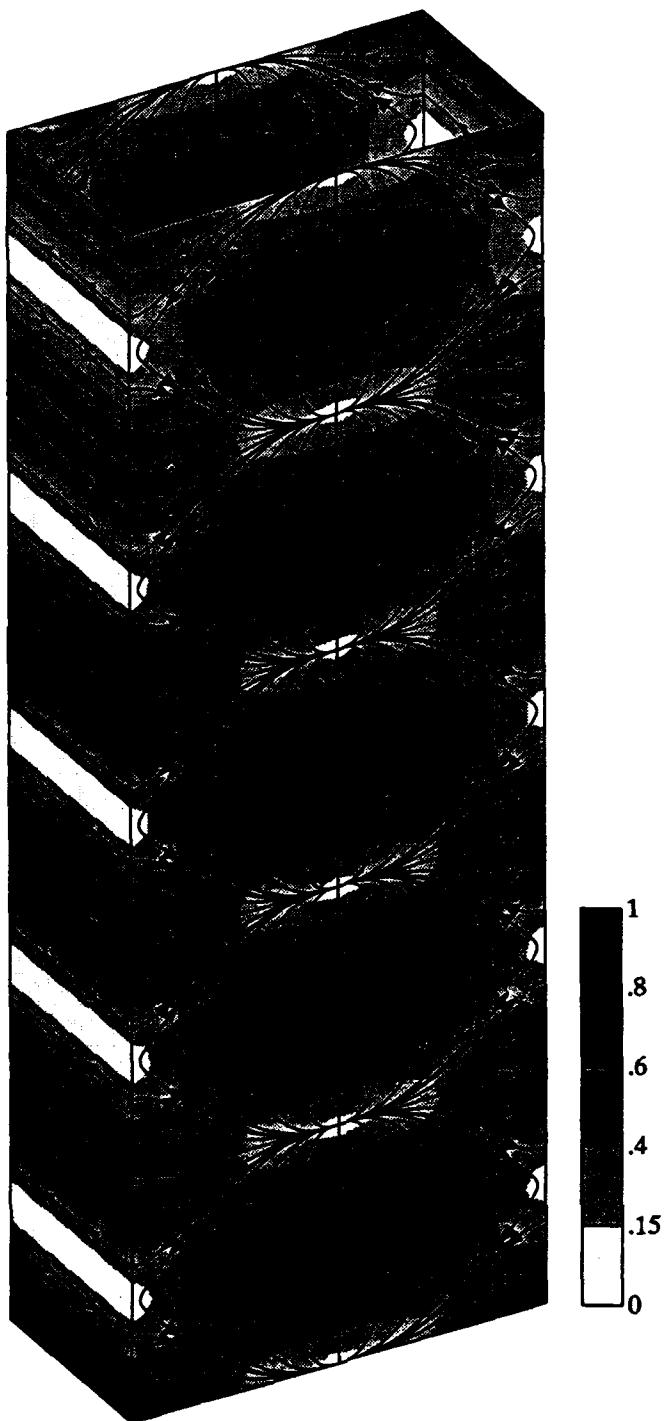


Fig. 4. Magnitude and Direction of  $TE_{10}$  Surface Current Plotted on Rectangular Waveguide

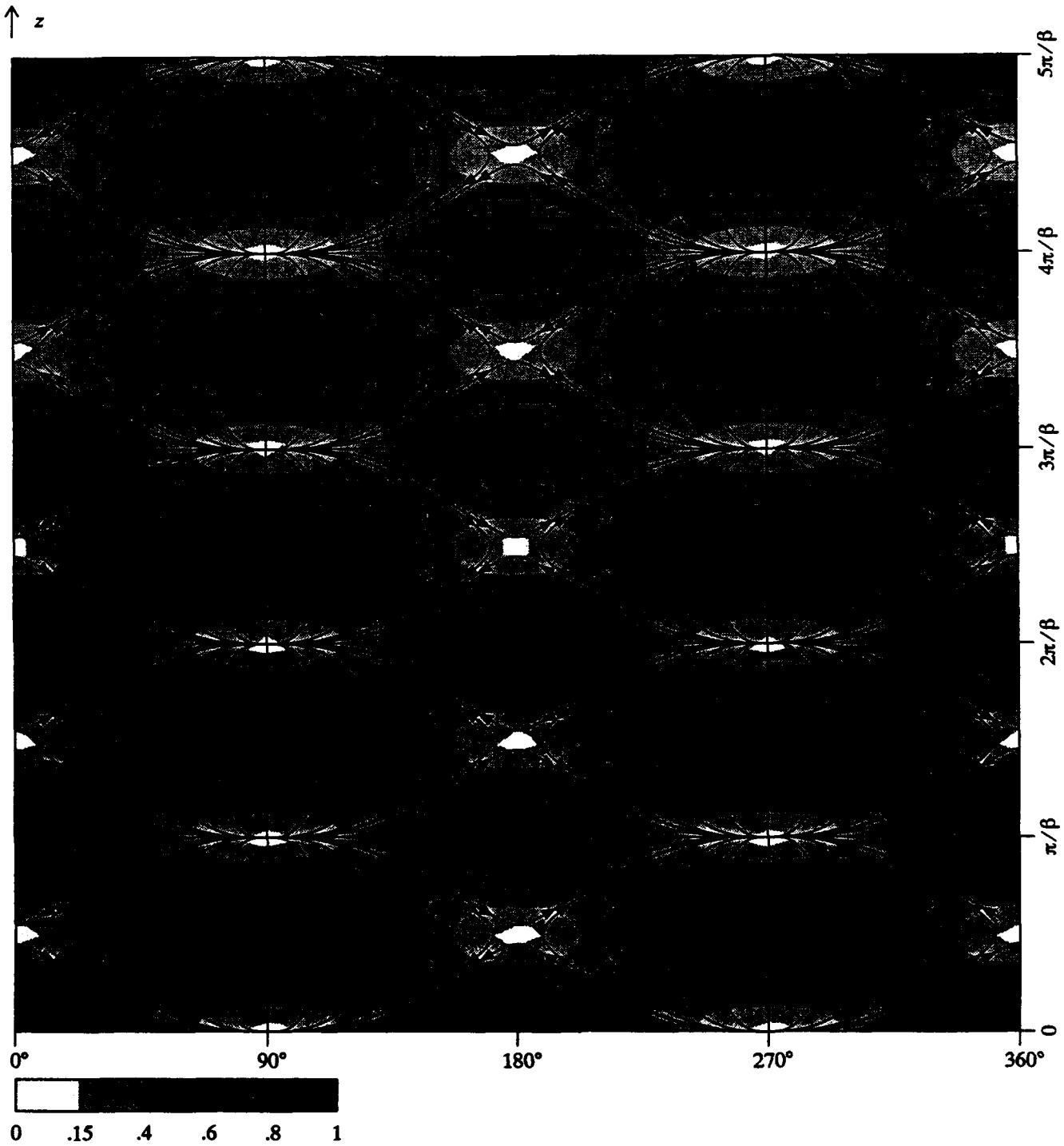


Fig. 5. Magnitude and Direction of  $TE_{11}$  Surface Current Plotted on Unfolded Circular Waveguide

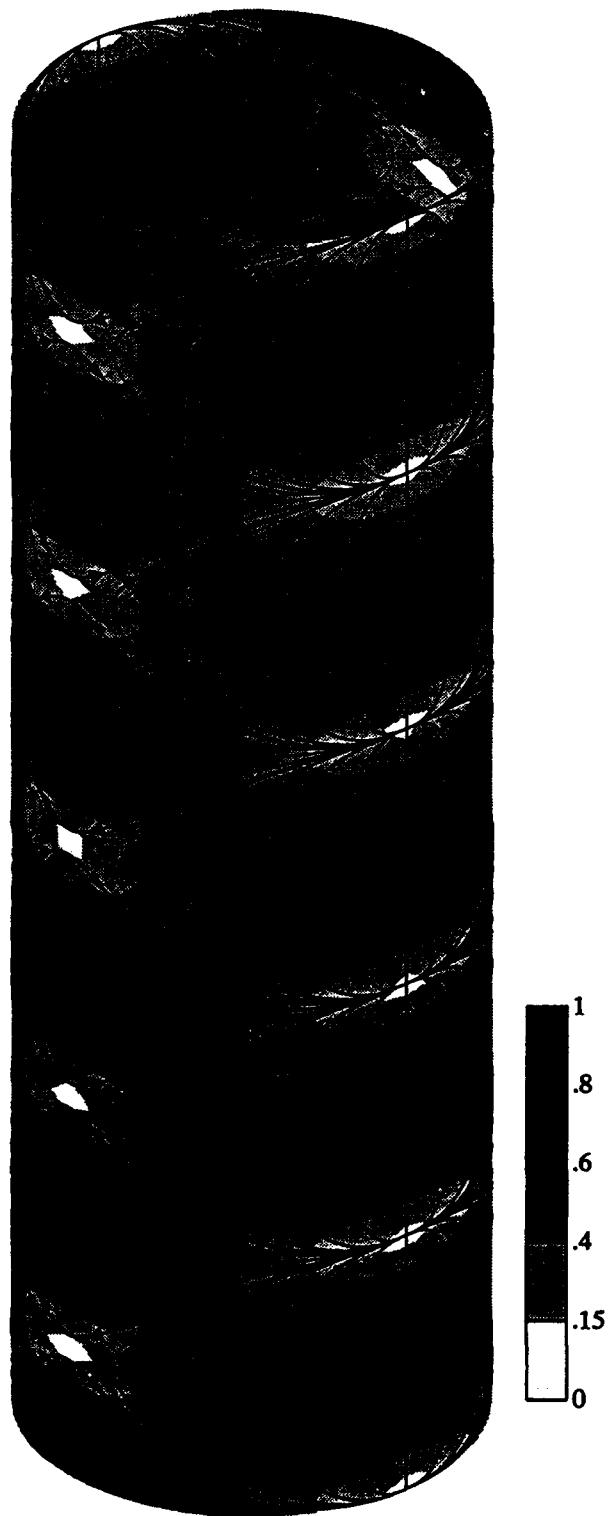


Fig. 6. Magnitude and Direction of TE<sub>11</sub> Surface Current Plotted on Circular Waveguide

## VI. CONCLUSION

An algorithm has been presented for plotting the magnitude and direction of the current induced on the inside surface of a rectangular or circular waveguide. The algorithm is capable of plotting either a two- or three-dimensional view of the surface-current field. The use of a triangularized mesh is consistent with the output generated by a finite-element solution. Test cases for the lower-order modes of a rectangular and circular waveguide yielded excellent results.

## REFERENCES

- [1] P. E. Moller and R. H. Macphie, "On the Graphical Representation of Electric Field Lines in Waveguide," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-33, pp. 187-192, Mar. 1985.
- [2] C. S. Lee, S. W. Lee, and S. L. Chuang, "Plot of Modal Field Distribution in Rectangular and Circular Waveguides," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-33, pp. 271-274, Mar. 1985.
- [3] D. Kajfez and J. A. Gerald, "Plotting Vector Fields with a Personal Computer," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-35, pp. 1069-1072, Nov. 1987.
- [4] K. W. Kark and R. Dill, "A General Theory on the Graphical Representation of Antenna-Radiation Fields," *IEEE Trans. Antennas Propagat.*, vol. AP-38, pp. 160-165, Feb. 1990.
- [5] A. A. Read, "Computers and Computer Graphics in the Teaching of Field Phenomena," *IEEE Trans. Educ.*, vol. E-33, pp. 95-103, Feb. 1990.
- [6] F. G. Stremler, S. A. Klein, F. F. Luo, and Y. Liao, "Numerical Solutions and Mapping of Electrostatic Fields using the Apple Macintosh Computer," *IEEE Trans. Educ.*, vol. E-33, pp. 104-110, Feb. 1990.
- [7] T. J. Peters, "A Fast Algorithm for Plotting Antenna and Scattering Patterns in Three Dimensions," The Aerospace Corp., Los Angeles, CA, Tech. Rep. TR-0091(6925-05)-6, August 1991.
- [8] T. J. Peters, "A Fast Algorithm for Plotting and Contour Filling Radiation Patterns in Three Dimensions," *IEEE Trans. Antennas Propagat.*, vol. AP-40, pp. 453-456, April. 1992.
- [9] A. E. Perry and M. S. Chong, "A Description of Eddying Motions and Flow Patterns using Critical Point Concepts," *Ann. Rev. Fluid Mechanics*, vol. 19, pp. 125-155, 1987.
- [10] J. L. Helman and L. Hessenlink, "Representation and Display of Vector Field Topology in Fluid Flow Data Sets," *Computer*, vol. 22, pp. 27-36, Aug. 1989.
- [11] J. L. Helman and L. Hessenlink, "Visualizing Vector Field Topology in Fluid Flows," *IEEE Comput. Graph. Appl.*, vol. 11, pp. 36-46, May. 1991.
- [12] G. M. Nielson, T. A. Foley, B. Hamann, and D. Lane, "Visualizing and Modeling Scattered Multivariate Data," *IEEE Comput. Graph. Appl.*, vol. 11, pp. 47-55, May. 1991.

## Appendix A: Vector-Line Differential Equation Solution

The slope of a line tangent to the vector field over a triangle determines a first-order nonlinear differential equation of the form

$$\frac{dy}{dx} = \frac{Ax + By + E}{Cx + Dy + F}. \quad (A1)$$

The solution of this differential equation is given by Davis [A1] and is presented below. The first step is to convert the equation to homogeneous form through the definitions

$$x = u + h \quad (A2)$$

$$y = v + g. \quad (A3)$$

Substituting these expressions into Eq. (A1) yields the homogeneous differential equation

$$\frac{dv}{du} = \frac{Au + Bv}{Cu + Dv} \quad (A3)$$

where it is assumed that  $h$  and  $g$  satisfy the linear equations

$$Ah + Bg = -E \quad (A4)$$

$$Ch + Dg = -F \quad (A5)$$

such that

$$h = \frac{BF - DE}{AD - BC} \quad (A6)$$

$$g = \frac{CE - AF}{AD - BC}. \quad (A7)$$

Assuming that  $u > 0$  is satisfied over the triangle, define the variable  $s = v/u$  and substitute into Eq. (A4) such that

$$\frac{dv}{du} = s + u \frac{ds}{du}. \quad (A8)$$

This equation can then be separated as

$$\frac{1}{u} du = \frac{C + Ds}{A + (B - C)s - Ds^2} ds \quad (A9)$$

to allow the integration of both sides, yielding an implicit function of the form

$$W(x, y) = 0. \quad (A10)$$

This function can be expressed as

$$W(x, y) = \ln(x - h) - L(x, y) + c_0 \quad (A11)$$

where

$$L(x, y) = \int \left[ \frac{C + Ds}{A + (B - C)s - Ds^2} \right] ds \quad (A12)$$

and  $c_0$  is an integration constant chosen to match the initial position of the vector.

The integral identity has the form given by [A2]:

$$\int \left[ \frac{C + Ds}{A + (B - C)s - Ds^2} \right] ds = -\frac{1}{2} \ln[A + (B - C)s - Ds^2] + H(s) \quad (A13)$$

where

$$H(s) = \begin{cases} \frac{B+C}{T} \tan^{-1} \left[ \frac{B-C-2Ds}{T} \right] & \Delta > 0 \\ \frac{B+C}{2T} \ln \left[ \frac{B-C-T-2Ds}{B-C+T-2Ds} \right] & \Delta < 0 \end{cases} \quad (A14)$$

and

$$\Delta = -4AD - (B - C)^2 \quad (A15)$$

$$T = \sqrt{|\Delta|}. \quad (A16)$$

This identity allows Eq. (A12) to be evaluated as

$$\begin{aligned} L(x, y) = & \ln(x - h) - \frac{1}{2} \ln[A(x - h)^2 + (B - C)(x - h)(y - g) - D(y - g)^2] \\ & + H(x, y) \end{aligned} \quad (A17)$$

where

$$H(x, y) = \begin{cases} \frac{B+C}{T} \tan^{-1} \left[ \frac{(B-C)(x-h)-2D(y-g)}{T(x-h)} \right] & \Delta > 0 \\ \frac{B+C}{2T} \ln \left[ \frac{(B-C-T)(x-h)-2D(y-g)}{(B-C+T)(x-h)-2D(y-g)} \right] & \Delta < 0 \end{cases} \quad (A18)$$

Thus, Eq. (A11) can be written as

$$W(x, y) = \frac{1}{2} \ln[A(x - h)^2 + (B - C)(x - h)(y - g) - D(y - g)^2] - H(x, y) + c_0. \quad (A19)$$

The implicit form of the equation of the line makes this solution fairly slow to plot a vector field. However, it can be a useful way to judge interactively the appropriateness of the step size. A viewer-selected triangle can be enlarged and the vector drawn using either the incremental method described previously, or the analytic solution. A comparison of both methods can indicate the correctness of the step size or the triangle grid size.

## APPENDIX A REFERENCES

- [A1] H. T. Davis, "*Introduction to Nonlinear Differential and Integral Equations*," (New York: Dover, 1962), pp. 38-45.
- [A2] I. S. Gradshteyn and I. M. Ryzhik, "*Table of Integrals, Series, and Products*," (New York: Academic Press, 1980), p. 57.

## Appendix B: Computer Programs

This appendix contains the programs and subroutines used to generate the graphics found in this report. All source code is written in standard Fortran 77 [B1]. The output graphic is generated as an encapsulated PostScript (EPS) file [B2]-[B4]. The program **WAVEDR** generates the geometry of the waveguide, as well as that of the node-to-triangle matrix and the triangle-to-node matrix. The program **UNFOLD** plots the surface-current vector field on an unfolded circular or rectangular waveguide. The program **FLDREC** plots the surface-current vector field directly on the outside surface of a rectangular waveguide. The program **FLDCIR** plots the surface-current vector field directly on the outside surface of a circular waveguide. Figure B.1 shows the interdependency of the programs and subroutines.

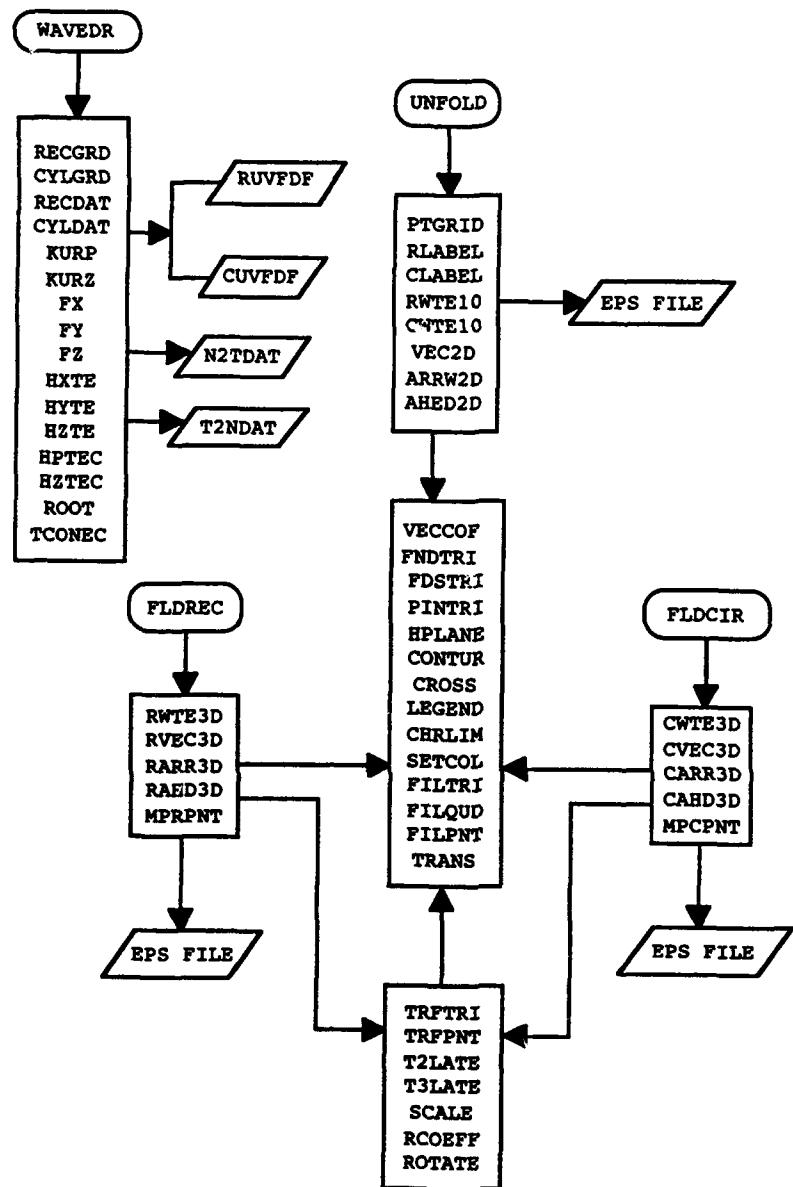


Fig. B.1 Program and Subroutine Dependencies

```

1      PROGRAM WAVEDR
2 C ****
3 C * THIS PROGRAM GENERATES THE DATA FILES FOR PLOTTING SURFACE   *
4 C * CURRENT MAGNITUDE AND DIRECTION ON A RECTANGULAR OR CIRCULAR   *
5 C * WAVEGUIDE AT TIME T=0.                                         *
6 C ****
7 C * TIMOTHY J. PETERS                               LAST UPDATED   *
8 C * THE AEROSPACE CORPORATION                      5/14/92        *
9 C * 2350 EAST EL SEGUNDO BOULEVARD                *
10 C * EL SEGUNDO, CA 90245                           *
11 C ****
12 C PARAMETER (MN=6000,MT=12000,MP=6)
13 C REAL*4 U(MN),V(MN),KC2
14 C INTEGER*4 N2T(MT,3),T2N(MN,MP)
15 C PI=.3141593E+01
16 C ****
17 C * INPUTS:                                         *
18 C *
19 C *      MN - MAXIMUM NUMBER OF NODES.             *
20 C *      MT - MAXIMUM NUMBER OF TRIANGLES.          *
21 C *      MP - MAXIMUM NUMBER OF TRIANGLES CONNECTED TO A NODE.   *
22 C *      IGE0 - IGE0=1 (RECTANGULAR) IGE0=2 (CIRCULAR) WAVEGUIDE.   *
23 C *      A - SIDE WIDTH -A/2 <=X<=A/2 FOR RECTANGULAR GUIDE OR   *
24 C *           RADIUS FOR CIRCULAR GUIDE (WAVELENGTHS).            *
25 C *      B - SIDE WIDTH -B/2 <=Y<=B/2 FOR RECTANGULAR GUIDE.       *
26 C *      KM - NUMBER OF LONGITUDINAL HALF CYCLES.               *
27 C *      M,N - MODE NUMBERS FOR TE_{MN} FOR RECTANGULAR GUIDE.    *
28 C *      N,M - MODE NUMBERS FOR TE_{NM} FOR CIRCULAR GUIDE.       *
29 C *      NUA - NUMBER OF POINTS ALONG X DIRECTION.              *
30 C *      NUB - NUMBER OF POINTS ALONG Y DIRECTION.              *
31 C ****
32 C IGE0=2
33 C IF (IGEO .EQ. 1) THEN
34 C ****
35 C * RECTANGULAR WAVEGUIDE.                                *
36 C ****
37 C A=1.0
38 C B=0.5
39 C M=1
40 C N=0
41 C KM=5
42 C NV=62
43 C NUA=27
44 C NUB=9
45 C KC2=(M*PI/A)*(M*PI/A)+(N*PI/B)*(N*PI/B)
46 C BETA=SQRT(4*PI*PI-KC2)
47 C C=KM*PI/BETA
48 C NU=2*(NUA+NUB)-3
49 C NN=NU*NV
50 C NT=2*(NU-1)*(NV-1)
51 C ****
52 C * GENERATE THE N2T COORDINATES AND THE CONNECTION MATRIX.   *
53 C *** . ****
54 C CALL RECGRD(A,B,C,NUA,NUB,NU,NV,MN,MT,U,V,N2T)
55 C ****
56 C * WRITE OUT THE COORDINATE DATA AND VECTOR FUNCTION DATA TO A  *
57 C * FILE.                                                 *
58 C ****
59 C CALL RECDAT(A,B,C,KM,NUA,NUB,NV,NN,MN,U,V,M,N)

```

```

60      ELSE
61          NU=45
62          NV=62
63          NN=NU*NV
64          NT=2*(NU-1)*(NV-1)
65          A=0.5
66          N=1
67          M=1
68          PNL=ROOT(N,M)
69          KC2=PNL*PNL/(A*A)
70          BETA=SQRT(4*PI*PI-KC2)
71          KM=5
72          C=KM*PI/BETA
73          CALL CYLGRD(A,C,NU,NV,MN,MT,U,V,N2T)
74 C          *****
75 C          * WRITE OUT THE COORDINATE DATA AND VECTOR FUNCTION DATA TO A *
76 C          * FILE. *
77 C          *****
78          CALL CYLD_T(A,C,KM,NU,NV,NN,MN,U,V,N,M)
79      END IF
80 C          *****
81 C          * WRITE OUT THE NODE TO TRIANGLE CONNECTION DATA. *
82 C          *****
83          OPEN(UNIT=1,FILE='N2TDAT')
84          WRITE(1,*) NT
85          DO 1 I=1,NT
86              WRITE(1,*) N2T(I,1),N2T(I,2),N2T(I,3)
87 1      CONTINUE
88          CLOSE(1)
89 C          *****
90 C          * GENERATE THE MATRIX OF TRIANGLES CONNECTED TO THE SAME N2T. *
91 C          *****
92          CALL TCONEC(MT,N2T,NT,MN,MP,T2N)
93          WRITE(*,*) 'TRIANGLE TO NODE CONNECTION MATRIX GENERATED'
94          OPEN(UNIT=1,FILE='T2NDAT')
95          WRITE(1,*) NN
96          DO 2 I=1,NN
97              WRITE(1,*) T2N(I,1),T2N(I,2),T2N(I,3),T2N(I,4),T2N(I,5),T2N(I,6)
98 2      CONTINUE
99          CLOSE(1)
100         END

1      SUBROUTINE RECGRD(A,B,C,NUA,NUB,NU,NV,MN,MT,U,V,N2T)
2 C          *****
3 C          * THIS SUBROUTINE GENERATES THE UNFOLDED COORDINATES OF A *
4 C          * SECTION OF RECTANGULAR WAVEGUIDE. THESE COORDINATES REPRESENT *
5 C          * THE VERTICES OF TRIANGLES. NODE TO TRIANGLE CONNECTION MATRIX *
6 C          * IS ALSO GENERATED. *
7 C          *****
8 C          * INPUTS: *
9 C          *
10 C          *   A - WIDTH OF WAVEGUIDE IN X DIRECTION -A/2 <=X<= A/2. *
11 C          *   B - WIDTH OF WAVEGUIDE IN Y DIRECTION -B/2 <=Y<= B/2. *
12 C          *   C - LENGTH OF WAVEGUIDE IN Z DIRECTION. *
13 C          *   NUA - NUMBER OF POINTS ALONG X DIRECTION. *
14 C          *   NUB - NUMBER OF POINTS ALONG Y DIRECTION. *
15 C          *   NU - TOTAL NUMBER OF U POINTS. *
16 C          *   NV - TOTAL NUMBER OF V POINTS. *
17 C          *   MN - MAXIMUM DIMENSION OF VECTORS U AND V. *

```

```

18 C * MT - MAXIMUM NUMBER OF TRIANGLES. *
19 C *
20 C * OUTPUTS: *
21 C *
22 C * U(MN) - COORDINATES OF EACH DATA POINT. *
23 C * V(MN) *
24 C * N2T(MT,3) MATRIX OF N2T NUMBERS CONNECTED TO EACH *
25 C * TRIANGLE. *
26 C ****
27 REAL*4 U(MN),V(MN)
28 INTEGER*4 N2T(MT,3)
29 PI=.3141593E+01
30 C ****
31 C * GENERATE THE N2T COORDINATES. *
32 C ****
33 DUA=A/(NUA-1)
34 DUB=B/(NUB-1)
35 DV=C/(NV-1)
36 C ****
37 C * SECTION 1 HORIZONTAL COORDINATE. *
38 C ****
39 K=0
40 DO 1 I=0,NUA-2
41     K=K+1
42     U(K)=-B-A+I*DUA
43 1 CONTINUE
44 C ****
45 C * SECTION 2 HORIZONTAL COORDINATE. *
46 C ****
47 DO 2 I=0,NUB-2
48     K=K+1
49     U(K)=-B+I*DUB
50 2 CONTINUE
51 C ****
52 C * SECTION 3 HORIZONTAL COORDINATE. *
53 C ****
54 DO 3 I=0,NUA-2
55     K=K+1
56     U(K)=I*DUA
57 3 CONTINUE
58 C ****
59 C * SECTION 4 HORIZONTAL COORDINATE. *
60 C ****
61 DO 4 I=0,NUB-1
62     K=K+1
63     U(K)=A+I*DUB
64 4 CONTINUE
65 C ****
66 C * LONGITUDINAL (VERTICAL) COORDINATE. *
67 C ****
68 K=0
69 DO 5 I=0,NV-1
70     K=K+1
71     V(K)=I*DV
72 5 CONTINUE
73 C ****
74 C * GENERATE THE CONNECTION MATRIX. *
75 C ****
76 K=0

```

```

77      N=0
78      DO 6 I=1,NU-1
79          DO 7 J=1,NV-1
80      N=N+1
81          K=K+1
82          N2T(K,1)=N
83          N2T(K,2)=N+1
84          N2T(K,3)=N+NV+1
85          K=K+1
86          N2T(K,1)=N
87          N2T(K,2)=N+NV
88          N2T(K,3)=N+NV+1
89      7    CONTINUE
90      N=N+1
91      6    CONTINUE
92      RETURN
93      END

1      SUBROUTINE CYLGRD(A,C,NU,NV,MN,MT,U,V,N2T)
2 C ****
3 C * THIS SUBROUTINE GENERATES THE UNFOLDED COORDINATES OF A *
4 C * SECTION OF RECTANGULAR WAVEGUIDE. THESE COORDINATES REPRESENT *
5 C * THE VERTICES OF TRIANGLES. NODE TO TRIANGLE CONNECTION MATRIX *
6 C * IS ALSO GENERATED.
7 C ****
8 C * INPUTS: *
9 C *
10 C *   A - WIDTH OF WAVEGUIDE IN X DIRECTION. *
11 C *   B - WIDTH OF WAVEGUIDE IN Y DIRECTION. *
12 C *   C - LENGTH OF WAVEGUIDE IN Z DIRECTION. *
13 C *   NUA - NUMBER OF POINTS ALONG X DIRECTION. *
14 C *   NUB - NUMBER OF POINTS ALONG Y DIRECTION. *
15 C *   NU - TOTAL NUMBER OF U POINTS. *
16 C *   NV - TOTAL NUMBER OF V POINTS. *
17 C *   MN - MAXIMUM DIMENSION OF VECTORS U AND V. *
18 C *   MT - MAXIMUM NUMBER OF TRIANGLES. *
19 C *
20 C * OUTPUTS: *
21 C *
22 C *   U(MN) - COORDINATES OF EACH DATA POINT. *
23 C *   V(MN) *
24 C *   N2T(MT,3) MATRIX OF N2T NUMBERS CONNECTED TO EACH *
25 C *   TRIANGLE.
26 C ****
27 REAL*4 U(MN),V(MN)
28 INTEGER*4 N2T(MT,3)
29 TP=.62831853E+01
30 C ****
31 C * GENERATE THE N2T COORDINATES. *
32 C ****
33 DU=TP*A/(NU-1)
34 DV=C/(NV-1)
35 C ****
36 C * SECTION 1 HORIZONTAL COORDINATE. *
37 C ****
38 K=0
39 DO 1 I=0,NU-1
40     K=K+1
41     U(K)=I*DU

```

```

42 1    CONTINUE
43 C ****
44 C * LONGITUDINAL (VERTICAL) COORDINATE. *
45 C ****
46 K=0
47 DO 5 I=0,NV-1
48     K=K+1
49     V(K)=I*DV
50 5    CONTINUE
51 C ****
52 C * GENERATE THE CONNECTION MATRIX. *
53 C ****
54 K=0
55 N=0
56 DO 6 I=1,NU-1
57     DO 7 J=1,NV-1
58     N=N+1
59     K=K+1
60     N2T(K,1)=N
61     N2T(K,2)=N+1
62     N2T(K,3)=N+NV+1
63     K=K+1
64     N2T(K,1)=N
65     N2T(K,2)=N+NV
66     N2T(K,3)=N+NV+1
67 7    CONTINUE
68     N=N+1
69 6    CONTINUE
70 RETURN
71 END

1    SUBROUTINE RECDAT(A,B,C,KM,NUA,NUB,NV,NN,MN,U,V,M,N)
2 C ****
3 C * THIS SUBROUTINE WRITES OUT THE COORDINATE AND FUNCTION DATA *
4 C * FOR A RECTANGULAR WAVEGUIDE TO A FILE. *
5 C ****
6 REAL*4 U(MN),V(MN)
7 DX=A/(NUA-1)
8 DY=B/(NUB-1)
9 DZ=C/(NV-1)
10 OPEN(UNIT=1,FILE='RUVFDF')
11 WRITE(1,*) A,B,C,KM,M,N
12 WRITE(1,*) NUA,NUB,NV
13 WRITE(1,*) NN
14 C ****
15 C * SIDE 1. *
16 C ****
17 Y=B/2.0
18 VX=0.0
19 VY=-1.0
20 VZ=0.0
21 K=0
22 DO 1 I=0,NUA-1
23     X=-A/2.0+I*DX
24     K=K+1
25     DO 2 J=0,NV-1
26         Z=J*DZ
27         CALL FZ(A,B,M,N,X,Y,Z,VX,VY,VZ,FZV)
28         CALL FX(A,B,M,N,X,Y,Z,VX,VY,VZ,FXV)

```

```

29      WRITE(1,*) U(K),V(J+1),FXV,FZV
30 2    CONTINUE
31 1    CONTINUE
32 C   ****
33 C   * SIDE 2.
34 C   ****
35     X=A/2.0
36     VX=-1.0
37     VY=0.0
38     VZ=0.0
39     DO 3 I=1,NUB-1
40       Y=B/2.0-I*DY
41       K=K+1
42     DO 4 J=0,NV-1
43       Z=J*DZ
44       CALL FZ(A,B,M,N,X,Y,Z,VX,VY,VZ,FZV)
45       CALL FY(A,B,M,N,X,Y,Z,VX,VY,VZ,FYV)
46       WRITE(1,*) U(K),V(J+1),-FYV,FZV
47 4    CONTINUE
48 3    CONTINUE
49 C   ****
50 C   * SIDE 3.
51 C   ****
52     Y=-B/2.0
53     VX=0.0
54     VY=1.0
55     VZ=0.0
56     DO 5 I=1,NUA-1
57       X=A/2.0-I*DX
58       K=K+1
59     DO 6 J=0,NV-1
60       Z=J*DZ
61       CALL FZ(A,B,M,N,X,Y,Z,VX,VY,VZ,FZV)
62       CALL FX(A,B,M,N,X,Y,Z,VX,VY,VZ,FXV)
63       WRITE(1,*) U(K),V(J+1),-FXV,FZV
64 6    CONTINUE
65 5    CONTINUE
66 C   ****
67 C   * SIDE 4.
68 C   ****
69     X=-A/2.0
70     VX=1.0
71     VY=0.0
72     VZ=0.0
73     DO 7 I=1,NUB-1
74       Y=-B/2.0+I*DY
75       K=K+1
76     DO 8 J=0,NV-1
77       Z=J*DZ
78       CALL FZ(A,B,M,N,X,Y,Z,VX,VY,VZ,FZV)
79       CALL FY(A,B,M,N,X,Y,Z,VX,VY,VZ,FYV)
80       WRITE(1,*) U(K),V(J+1),FYV,FZV
81 8    CONTINUE
82 7    CONTINUE
83    CLOSE(1)
84    RETURN
85    END

1    SUBROUTINE CYLDAT(A,C,KM,NU,NV,NN,MN,U,V,N,M)

```

```

2 C ****
3 C * THIS SUBROUTINE WRITES OUT THE COORDINATE AND FUNCTION DATA *
4 C * FOR A CIRCULAR WAVEGUIDE TO A FILE. *
5 C ****
6 REAL*4 U(MN),V(MN)
7 TP=.628318531E+01
8 DP=TP/(NU-1)
9 DZ=C/(NV-1)
10 OPEN(UNIT=1,FILE='CUVFDF')
11 WRITE(1,*) A,C,KM,N,M
12 WRITE(1,*) NU,NV
13 WRITE(1,*) NN
14 C ****
15 C * SIDE 1. *
16 C ****
17 R=A
18 K=0
19 DO 1 I=0,NU-1
20 P=I*DP
21 K=K+1
22 DO 2 J=0,NV-1
23 Z=J*DZ
24 CALL KURP(A,N,M,R,P,Z,FPV)
25 CALL KURZ(A,N,M,R,P,Z,FZV)
26 WRITE(1,*) U(K),V(J+1),FPV,FZV
27 2 CONTINUE
28 1 CONTINUE
29 CLOSE(1)
30 RETURN
31 END

1 SUBROUTINE KURP(A,N,L,R,P,Z,F)
2 ****
3 C * THIS SUBROUTINE COMPUTES THE SURFACE CURRENT FLOWING IN THE X *
4 C * DIRECTION. *
5 C ****
6 CALL HZTEC(A,N,L,R,P,Z,HZ)
7 F=HZ
8 RETURN
9 END

1 SUBROUTINE KURZ(A,N,L,R,P,Z,F)
2 ****
3 C * THIS SUBROUTINE COMPUTES THE SURFACE CURRENT FLOWING IN THE X *
4 C * DIRECTION. *
5 C ****
6 CALL HPTEC(A,N,L,R,P,Z,HP)
7 F=-HP
8 RETURN
9 END

1 SUBROUTINE FX(A,B,M,N,X,Y,Z,VX,VY,VZ,F)
2 ****
3 C * THIS SUBROUTINE COMPUTES THE SURFACE CURRENT FLOWING IN THE X *
4 C * DIRECTION. *
5 C ****
6 CALL HZTE(X,Y,Z,A,B,M,N,HZ)
7 CALL HYTE(X,Y,Z,A,B,M,N,HY)
8 F=VY*HZ-VZ*HY

```

```

9      RETURN
10     END

1      SUBROUTINE FY(A,B,M,N,X,Y,Z,VX,VY,VZ,F)
2 C      ****
3 C      * THIS SUBROUTINE COMPUTES THE SURFACE CURRENT FLOWING IN THE Y *
4 C      * DIRECTION. *
5 C      ****
6      CALL HXTE(X,Y,Z,A,B,M,N,HX)
7      CALL HZTE(X,Y,Z,A,B,M,N,HZ)
8      F=VZ*HX-VX*HZ
9      RETURN
10     END

1      SUBROUTINE FZ(A,B,M,N,X,Y,Z,VX,VY,VZ,F)
2 C      ****
3 C      * THIS SUBROUTINE COMPUTES THE SURFACE CURRENT FLOWING IN THE Z *
4 C      * DIRECTION. *
5 C      ****
6      CALL HYTE(X,Y,Z,A,B,M,N,HY)
7      CALL HXTE(X,Y,Z,A,B,M,N,HX)
8      F=VX*HY-VY*HX
9      RETURN
10     END

1      SUBROUTINE HXTE(X,Y,Z,A,B,M,N,HX)
2 C      ****
3 C      * THIS SUBROUTINE COMPUTES THE X COMPONENT OF THE MAGNETIC FIELD *
4 C      * ON THE SURFACE OF A RECTANGULAR WAVEGUIDE FOR A TE MODE. *
5 C      ****
6      PI=.3141593E+01
7      TP2=.394784176E+02
8      ARGX=(M*PI/A)*(X-A/2.0)
9      ARGY=(N*PI/B)*(Y-B/2.0)
10     H2=(M*PI/A)*(M*PI/A)+(N*PI/B)*(N*PI/B)
11     BETA=SQRT(TP2-H2)
12     ARGZ=-BETA*Z
13     H2=(M*PI/A)*(M*PI/A)+(N*PI/B)*(N*PI/B)
14     HX=(-M*PI*BETA/(A*H2))*SIN(ARGX)*COS(ARGY)*SIN(ARGZ)
15     RETURN
16     END

1      SUBROUTINE HYTE(X,Y,Z,A,B,M,N,HY)
2 C      ****
3 C      * THIS SUBROUTINE COMPUTES THE X COMPONENT OF THE MAGNETIC FIELD *
4 C      * ON THE SURFACE OF A RECTANGULAR WAVEGUIDE FOR A TE MODE. *
5 C      ****
6      PI=.3141593E+01
7      TP2=.394784176E+02
8      ARGX=(M*PI/A)*(X-A/2.0)
9      ARGY=(N*PI/B)*(Y-B/2.0)
10     H2=(M*PI/A)*(M*PI/A)+(N*PI/B)*(N*PI/B)
11     BETA=SQRT(TP2-H2)
12     ARGZ=-BETA*Z
13     HY=(-N*PI*BETA/(B*H2))*COS(ARGX)*SIN(ARGY)*SIN(ARGZ)
14     RETURN
15     END

1      SUBROUTINE HZTE(X,Y,Z,A,B,M,N,HZ)

```

```

2 C ****
3 C * THIS SUBROUTINE COMPUTES THE Z COMPONENT OF THE MAGNETIC FIELD *
4 C * ON THE SURFACE OF A RECTANGULAR WAVEGUIDE FOR A TE MODE. *
5 C ****
6 PI=.3141593E+01
7 TP2=.394784176E+02
8 ARGX=(M*PI/A)*(X-A/2.0)
9 ARGY=(N*PI/B)*(Y-B/2.0)
10 H2=(M*PI/A)*(M*PI/A)+(N*PI/B)*(N*PI/B)
11 BETA=SQRT(TP2-H2)
12 ARGZ=-BETA*Z
13 HZ=COS(ARGX)*COS(ARGY)*COS(ARGZ)
14 RETURN
15 END

1 SUBROUTINE HPTEC(A,N,L,R,P,Z,HP)
2 ****
3 C * THIS SUBROUTINE COMPUTES THE PHI COMPONENT OF THE MAGNETIC *
4 C * FIELD ON THE SURFACE OF A CIRCULAR WAVEGUIDE FOR A TE_{NL} *
5 C * MODE AT TIME T=0. NOTE THAT THE BESSSEL FUNCTION IS NOT *
6 C * INCLUDED SINCE IT WILL DROP OUT WHEN THE CURRENT IS NORMALIZED. *
7 C ****
8 REAL KC
9 PI=.3141593E+01
10 TP2=.394784176E+02
11 PNL=ROOT(N,L)
12 KC=PNL/A
13 BETA=SQRT(TP2-KC*I_0)
14 ARGZ=-BETA*Z
15 HP=(-N*TP2/(KC*KC*BETA*R))*SIN(N*P)*SIN(ARGZ)
16 RETURN
17 END

1 SUBROUTINE HZTEC(A,N,L,R,P,Z,HZ)
2 ****
3 C * THIS SUBROUTINE COMPUTES THE Z COMPONENT OF THE MAGNETIC FIELD *
4 C * ON THE SURFACE OF A CIRCULAR WAVEGUIDE FOR A TE_{NL} MODE. *
5 C * NOTE THAT THE BESSSEL FUNCTION IS NOT INCLUDED SINCE IT WILL *
6 C * DROP OUT WHEN THE CURRENT IS NORMALIZED. *
7 C ****
8 REAL KC
9 PI=.3141593E+01
10 TP2=.394784176E+02
11 PNL=ROOT(N,L)
12 KC=PNL/A
13 BETA=SQRT(TP2-KC*KC)
14 ARGZ=-BETA*Z
15 HZ=COS(N*P)*COS(ARGZ)
16 RETURN
17 END

1 FUNCTION ROOT(N,L)
2 ****
3 C * THIS FUNCTION RETURNS THE LTH ROOT OF THE FIRST *
4 C * DERIVATIVE OF A BESSSEL FUNCTION OF ORDER N. *
5 C ****
6 REAL*4 PNL(0:2,4)
7 PNL(0,1)=3.832
8 PNL(0,2)=7.016

```

```

9      PNL(0,3)=10.174
10     PNL(0,4)=13.324
11     PNL(1,1)=1.841
12     PNL(1,2)=5.331
13     PNL(1,3)=8.536
14     PNL(1,4)=11.706
15     PNL(2,1)=3.054
16     PNL(2,2)=6.706
17     PNL(2,3)=9.970
18     PNL(2,4)=13.170
19     ROOT=PNL(N,L)
20     RETURN
21     END

1      SUBROUTINE TCONEC(MT,N2T,NT,MN,MP,T2N)
2 C
3 C      * THIS SUBROUTINE GENERATES A MATRIX WITH GIVES ALL THE *
4 C      * TRIANGLES CONNECTED TO EACH N2T. *
5 C
6 C      * INPUTS: *
7 C
8 C      *      MT - MAXIMUM NUMBER OF TRIANGLES. *
9 C      *      N2T(MT,3)   MATRIX OF NODE NUMBERS CONNECTED TO EACH *
10 C      *                  TRIANGLE. *
11 C      *      MN - MAXIMUM NUMBER OF NODES. *
12 C      *      MP - MAXIMUM NUMBER OF POINTS CONNECTED TO EACH TRIANGLE. *
13 C
14 C      * OUTPUT: *
15 C
16 C      *      T2N(MN,MP)  MATRIX OF TRIANGLE NUMBERS CONNECTED TO EACH *
17 C      *                  NODE. *
18 C
19 C      INTEGER*4 N2T(MT,3),T2N(MN,MP)
20 C
21 C      * INITIALIZE THE MATRIX. *
22 C
23 DO 1 I=1,MN
24     DO 2 J=1,MP
25         T2N(I,J)=0
26     2 CONTINUE
27     1 CONTINUE
28 C
29 C      * FILL IN NON ZERO ENTRIES. *
30 C
31 DO 3 I=1,NT
32     DO 4 J=1,3
33         IFLAG=0
34         DO 5 K=1,MP
35             IF ((T2N(N2T(I,J),K) .EQ. 0) .AND. (IFLAG .EQ. 0)) THEN
36                 T2N(N2T(I,J),K)=I
37                 IFLAG=1
38             ELSE
39                 END IF
40         5 CONTINUE
41         4 CONTINUE
42     3 CONTINUE
43     RETURN
44     END

```

```

1      PROGRAM UNFOLD
2 C ****
3 C * THIS PROGRAM GENERATES A VECTOR PLOT OF A TRIANGULARIZED      *
4 C * UNFOLDED RECTANGULAR WAVEGUIDE.                                *
5 C ****
6 C * TIMOTHY J. PETERS                               LAST UPDATED   *
7 C * THE AEROSPACE CORPORATION                      5/12/92        *
8 C * 2350 EAST EL SEGUNDO BOULEVARD                 *
9 C * EL SEGUNDO, CA 90245                           *
10 C ****
11 C PARAMETER (MN=6000,MT=12000,MP=6,NC=5,NL=6)
12 C REAL*4 X(MN),Y(MN),FX(MN),FY(MN),PHI(50),CA(NC),CB(NC)
13 C REAL*8 ARG
14 C INTEGER LC(NC)
15 C CHARACTER*30 LABEL(NL)
16 C INTEGER*4 N2T(MT,3),T2N(MN,MP)
17 C PI=.3141593E+01
18 C TP=.6283153E+01
19 C RAD=.17453293E-01
20 C ****
21 C * INPUTS:                                         *
22 C *
23 C *      MN    - MAXIMUM NUMBER OF NODES.          *
24 C *      MT    - MAXIMUM NUMBER OF TRIANGLES.       *
25 C *      IP    - MAXIMUM NUMBER OF TRIANGLES CONNECTED TO A NODE. *
26 C *      IGE0  - IGE0=1 (RECTANGULAR) IGE0=2 (CIRCULAR) WAVEGUIDE. *
27 C *      A     - SIDE WIDTH -A/2 <=X<=A/2 FOR RECTANGULAR GUIDE OR *
28 C *                  RADIUS FOR CIRCULAR GUIDE (WAVELENGTHS). *
29 C *      B     - SIDE WIDTH -B/2 <=Y<=B/2 FOR RECTANGULAR GUIDE. *
30 C *      KM   - NUMBER OF LONGITUDINAL HALF CYCLES. *
31 C *      NUA  - NUMBER OF POINTS ALONG X DIRECTION. *
32 C *      NUB  - NUMBER OF POINTS ALONG Y DIRECTION. *
33 C ****
34 C ****
35 C * READ THE COORDINATE DATA AND VECTOR FUNCTION DATA FROM A FILE. *
36 C ****
37 C IGE0=2
38 C IF (IGEO .EQ. 1) THEN
39 C *
40 C * RECTANGULAR WAVEGUIDE.                                     *
41 C ****
42 C OPEN(UNIT=1,FILE='RUVFDF')
43 C READ(1,*) A,B,C,KM,M,N
44 C READ(1,*) NUA,NUB,NV
45 C READ(1,*) NN
46 C VMAX=0.0
47 C DO 1 I=1,NN
48 C     READ(1,*) X(I),Y(I),FX(I),FY(I)
49 C VMAG=SQRT(FX(I)*FX(I)+FY(I)*FY(I))
50 C     IF (VMAG .GT. VMAX) THEN
51 C         VMAX=VMAG
52 C     ELSE
53 C     END IF
54 C 1 CONTINUE
55 C CLOSE(1)
56 C WRITE(*,*) NN,' DATA POINTS READ IN'
57 C ELSE
58 C *
59 C * CIRCULAR WAVEGUIDE.                                     *

```

```

60 C ****
61 OPEN(UNIT=1,FILE='CUVFDF')
62 READ(1,*) A,C,KM,N,M
63 READ(1,*) NU,NV
64 READ(1,*) NN
65 VMAX=0.0
66 DO 2 I=1,NN
67   READ(1,*) X(I),Y(I),FX(I),FY(I)
68   VMAG=SQRT(FX(I)*FX(I)+FY(I)*FY(I))
69   IF (VMAG .GT. VMAX) THEN
70     VMAX=VMAG
71   ELSE
72     END IF
73 2 CONTINUE
74 CLOSE(1)
75 WRITE(*,*) NN,' DATA POINTS READ IN'
76 END IF
77 C ****
78 C * NORMALIZE THE VECTOR COMPONENTS SO THAT THE MAXIMUM MAGNITUDE *
79 C * IS 1.0. *
80 C ****
81 DO 3 I=1,NN
82   FX(I)=FX(I)/VMAX
83   FY(I)=FY(I)/VMAX
84 3 CONTINUE
85 C ****
86 C * READ THE NODE TO TRIANGLE CONNECTION MATRIX. *
87 C ****
88 OPEN(UNIT=1,FILE='N2TDAT')
89 READ(1,*) NT
90 DO 4 I=1,NT
91   READ(1,*) N2T(I,1),N2T(I,2),N2T(I,3)
92 4 CONTINUE
93 CLOSE(1)
94 WRITE(*,*) 'NODE TO TRIANGLE DATA READ IN'
95 C ****
96 C * READ THE TRIANGLE TO NODE CONNECTION MATRIX. *
97 C ****
98 OPEN(UNIT=1,FILE='T2NDAT')
99 READ(1,*) NN
100 DO 5 I=1,NN
101   READ(1,*) (T2N(I,J),J=1,MP)
102 5 CONTINUE
103 CLOSE(1)
104 WRITE(*,*) 'TRIANGLE TO NODE DATA READ IN'
105 C ****
106 C * COMPUTE THE CENTER OF THE PLOT REGION. *
107 C ****
108 IF (IGEO .EQ. 1) THEN
109   XMIN=-B-A
110   XMAX=A+B
111 ELSE
112   XMIN=0.0
113   XMAX=TP*A
114 END IF
115 YMIN=0.0
116 YMAX=C
117 XC=(XMIN+XMAX)/2.0
118 YC=(YMIN+YMAX)/2.0

```

```

119 C ****
120 C * COMPUTE THE CENTER OF THE PRINTING DEVICE PAGE. *
121 C ****
122 X0=72*8.5/2.0
123 Y0=72*11.0/2.0+72.0
124 C ****
125 C * COMPUTE THE SCALE CONSTANTS. *
126 C ****
127 GS=150.0
128 BX=X0-GS*XC
129 BY=Y0-GS*YC
130 C ****
131 C * SET THE GRAPHIC BOUNDING BOX. *
132 C ****
133 WX=(XMAX-XMIN)
134 WY=(YMAX-YMIN)
135 WIDTH=GS*WX
136 HEIGHT=GS*WY
137 UA=X0-WIDTH/2.0
138 UB=X0+WIDTH/2.0+21.0
139 VA=Y0-HEIGHT/2.0-53.0
140 VB=Y0+HEIGHT/2.0+25.0
141 C ****
142 C * COMPUTE THE BOUNDING BOX FOR THE POSTSCRIPT IN THE DEFAULT *
143 C * COORDINATE SYSTEM. *
144 C ****
145 OPEN(UNIT=1,FILE='Waveguide.ps')
146 REWIND 1
147 WRITE(1,*) '%!PS-Adobe-1.0'
148 WRITE(1,*) '%%Creator: Timothy J. Peters'
149 WRITE(1,*) '%%Title: Unfolded Waveguide'
150 WRITE(1,*) '%%CreationDate: 5-12-92'
151 WRITE(1,100) '%%BoundingBox:',INT(UA),INT(VA),INT(UB),INT(VB)
152 100 FORMAT(A14,4(1X,I3))
153 WRITE(1,*) '%%EndComments'
154 WRITE(1,*) '/dot2 {2 0 360 arc 0 setgray fill stroke} def'
155 WRITE(1,*) '/dot3 {3 0 360 arc 0 setgray fill stroke} def'
156 WRITE(1,*) '/dot4 {4 0 360 arc 0 setgray fill stroke} def'
157 WRITE(1,*) '/black {0 0 0 setrgbcolor} def'
158 WRITE(1,*) '/white {1 1 1 setrgbcolor} def'
159 WRITE(1,*) '/gray-lt {0.92 0.92 0.92 setrgbcolor} def'
160 WRITE(1,*) '/gray-lt-med {0.65 0.65 0.65 setrgbcolor} def'
161 WRITE(1,*) '/gray {0.45 0.45 0.45 setrgbcolor} def'
162 WRITE(1,*) '/gray-dk-med {0.3 0.3 0.3 setrgbcolor} def'
163 WRITE(1,*) '/gray-dk {0.14 0.14 0.14 setrgbcolor} def'
164 WRITE(1,*) '/red {1 0 0 setrgbcolor} def'
165 WRITE(1,*) '/magenta {1 0 1 setrgbcolor} def'
166 WRITE(1,*) '/green {0 1 0 setrgbcolor} def'
167 WRITE(1,*) '/blue {0 0 1 setrgbcolor} def'
168 WRITE(1,*) '/cyan {0 1 1 setrgbcolor} def'
169 WRITE(1,*) '/yellow {1 1 0 setrgbcolor} def'
170 WRITE(1,*) '/orange {1 0.5 0 setrgbcolor} def'
171 WRITE(1,*) '/brown {0.5 0.5 0 setrgbcolor} def'
172 WRITE(1,*) '/kakhi {0.5 1 0 setrgbcolor} def'
173 WRITE(1,*) '/blue-lt {0.5 1 1 setrgbcolor} def'
174 WRITE(1,*) '/green-lt {0.5 1 0.5 setrgbcolor} def'
175 WRITE(1,*) '/green-blue {0 1 0.5 setrgbcolor} def'
176 WRITE(1,*) '/purple {0.6 0 1.0 setrgbcolor} def'
177 WRITE(1,*) '/filtri {moveto lineto lineto'

```

```

178      WRITE(1,*), 'closepath fill stroke} def'
179      WRITE(1,*), '/filqud {moveto lineto lineto lineto closepath fill'
180      WRITE(1,*), '                      stroke} def'
181      WRITE(1,*), '/filpnt {moveto lineto lineto lineto'
182      WRITE(1,*), '                      lineto closepath fill stroke} def'
183      WRITE(1,*), '%EndProlog'
184      WRITE(1,*), '1 setlinejoin .4 setlinewidth'
185 C ****
186 C * ASSIGN CONTOUR VALUES AND LABELS.
187 C ****
188 CA(1)=0.0
189 CB(1)=0.15
190 CA(2)=0.15
191 CB(2)=0.4
192 CA(3)=0.4
193 CB(3)=0.6
194 CA(4)=0.6
195 CB(4)=0.8
196 CA(5)=0.8
197 CB(5)=1.0
198 LABEL(1)='0'
199 LABEL(2)='15'
200 LABEL(3)='4'
201 LABEL(4)='6'
202 LABEL(5)='8'
203 LABEL(6)='1'
204 C ****
205 C * ASSIGN THE COLORS. SEE SUBROUTINE SETCOL FOR COLOR CHOICES. *
206 C ****
207 C GRAY SCALE
208 LC(1)=2
209 LC(2)=3
210 LC(3)=4
211 LC(4)=5
212 LC(5)=6
213 C COLOR
214 C   LC(1)=18
215 C   LC(2)=14
216 C   LC(3)=12
217 C   LC(4)=16
218 C   LC(5)=13
219 C ****
220 C * DRAW THE CONTOUR PLOT.
221 C ****
222 WRITE(1,*), 'gsave'
223 DO 6 I=1,NT
224   N1=N2T(I,1)
225   N2=N2T(I,2)
226   N3=N2T(I,3)
227   X1=X(N1)
228   Y1=Y(N1)
229   X2=X(N2)
230   Y2=Y(N2)
231   X3=X(N3)
232   Y3=Y(N3)
233   U1=TRANS(GS,BX,X1)
234   V1=TRANS(GS,BY,Y1)
235   U2=TRANS(GS,BX,X2)
236   V2=TRANS(GS,BY,Y2)

```

```

237      U3=TRANS(GS,BX,X3)
238      V3=TRANS(GS,BY,Y3)
239      F1=SQRT(FX(N1)*FX(N1)+FY(N1)*FY(N1))
240      F2=SQRT(FX(N2)*FX(N2)+FY(N2)*FY(N2))
241      F3=SQRT(FX(N3)*FX(N3)+FY(N3)*FY(N3))
242      CALL CONTUR(U1,V1,U2,V2,U3,V3,F1,F2,F3,CA,CB,NC,LC)
243 6   CONTINUE
244      WRITE(1,*) 'grestore'
245 C ****
246 C * DRAW THE GRAPH FRAME. *
247 C ****
248      WRITE(1,*) 'gsave 0.5 setlinewidth 0 setgray'
249      WRITE(1,*) TRANS(GS,BX,XMIN),TRANS(GS,BY,YMIN), ' moveto'
250      WRITE(1,*) TRANS(GS,BX,XMAX),TRANS(GS,BY,YMIN), ' lineto'
251      WRITE(1,*) TRANS(GS,BX,XMAX),TRANS(GS,BY,YMAX), ' lineto'
252      WRITE(1,*) TRANS(GS,BX,XMIN),TRANS(GS,BY,YMAX), ' lineto'
253      WRITE(1,*) 'closepath stroke grestore'
254 C ****
255 C * DRAW THE EPS BOUNDING BOX. *
256 C ****
257 C      WRITE(1,*) 'gsave 0.2 setlinewidth'
258 C      WRITE(1,*) UA,VA,' moveto'
259 C      WRITE(1,*) UB,VA,' lineto'
260 C      WRITE(1,*) UB,VB,' lineto'
261 C      WRITE(1,*) UA,VB,' lineto'
262 C      WRITE(1,*) 'closepath stroke grestore'
263 C ****
264 C * LABEL EACH SECTION. *
265 C ****
266 IF (IGEO .EQ. 1) THEN
267 C ****
268 C * RECTANGULAR. *
269 C ****
270      CALL RLABEL(XMIN,XMAX,YMIN,YMAX,GS,BX,BY,A,B,KM)
271      SL=30.0
272      SH=15.0
273      DDS=0.0
274      ITYPE=2
275      US=TRANS(GS,BX,XMIN)
276      VS=TRANS(GS,BY,YMIN)-2.5*SH
277      CALL LEGEND(US,VS,NC,NL,LC,LABEL,SL,SH,DDS,ITYPE)
278 ELSE
279      KM=5
280      CALL CLABEL(XMIN,XMAX,YMIN,YMAX,GS,BX,BY,A,KM)
281      SL=30.0
282      SH=15.0
283      DDS=0.0
284      ITYPE=2
285      US=TRANS(GS,BX,XMIN)
286      VS=TRANS(GS,BY,YMIN)-2.5*SH
287      CALL LEGEND(US,VS,NC,NL,LC,LABEL,SL,SH,DDS,ITYPE)
288 END IF
289 C ****
290 C * IF REQUESTED SHOW THE TRIANGULAR GRID. *
291 C ****
292 C      CALL PTGRID(MN,MT,N'',N2T,GS,BX,BY,X,Y)
293 C ****
294 C * DRAW THE APPROPRIATE VECTOR ROUTINE. *
295 C ****

```

```

296      BANG=20.0
297      S=7.0
298      MNAR=1
299      DS=.003
300      IF (IGEO .EQ. 1) THEN
301
302          CALL RWTE10(MN,MT,MP,NT,N2T,T2N,X,Y,A,B,KM,M,N,XMIN,XMAX
303          & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
304      ELSE
305          CALL CWTE10(MN,MT,MP,NT,N2T,T2N,X,Y,A,C,KM,N,M,XMIN,XMAX
306          & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
307      END IF
308 C      ****
309 C      * DRAW THE PAGE.
310 C      ****
311      WRITE(1,*) 'showpage'
312      CLOSE(1)
313      END
314
315      SUBROUTINE PTGRID(MN,MT,NT,N2T,GS,BX,BY,X,Y)
316      ****
317      * THIS SUBROUTINE DRAWS THE TRIANGULAR GRID.
318      ****
319      REAL*4 X(MN),Y(MN)
320      INTEGER N2T(MT,3)
321
322      * LOOP THROUGH EACH TRIANGLE AND DRAW THE PERIMETER.
323      ****
324      WRITE(1,*) 'gsave o setgray 0.3 setlinewidth '
325      DO 1 I=1,NT
326          X1=X(N2T(I,1))
327          Y1=Y(N2T(I,1))
328          X2=X(N2T(I,2))
329          Y2=Y(N2T(I,2))
330          X3=X(N2T(I,3))
331          Y3=Y(N2T(I,3))
332          U1=TRANS(GS,BX,X1)
333          V1=TRANS(GS,BY,Y1)
334          U2=TRANS(GS,BX,X2)
335          V2=TRANS(GS,BY,Y2)
336          U3=TRANS(GS,BX,X3)
337          V3=TRANS(GS,BY,Y3)
338          CALL MOVETO(U1,V1)
339          CALL LINETO(U2,V2)
340          CALL MOVETO(U2,V2)
341          CALL LINETO(U3,V3)
342          CALL MOVETO(U3,V3)
343          CALL LINETO(U1,V1)
344 1      CONTINUE
345      WRITE(1,*) 'grestore'
346      RETURN
347      END
348
349      SUBROUTINE RLABEL(XMIN,XMAX,YMIN,YMAX,GS,BX,BY,A,B,KM)
350      ****
351      * THIS SUBROUTINE DRAWS THE LABELS FOR THE RECTANGULAR GUIDE.
352      ****
353      ****
354      * SECTION LINES.
355      ****

```

```

7 C ****
8 UD=TRANS(GS,BX,XMIN+A)
9 VD=TRANS(GS,BY,YMIN)
10 WRITE(1,*) UD,VD,' moveto'
11 UD=TRANS(GS,BX,XMIN+A)
12 VD=TRANS(GS,BY,YMAX)
13 WRITE(1,*) UD,VD,' lineto stroke'
14 UD=TRANS(GS,BX,XMIN+A+B)
15 VD=TRANS(GS,BY,YMIN)
16 WRITE(1,*) UD,VD,' moveto'
17 UD=TRANS(GS,BX,XMIN+A+B)
18 VD=TRANS(GS,BY,YMAX)
19 WRITE(1,*) UD,VD,' lineto stroke'
20 UD=TRANS(GS,BX,XMAX-B)
21 VD=TRANS(GS,BY,YMIN)
22 WRITE(1,*) UD,VD,' moveto'
23 UD=TRANS(GS,BX,XMAX-B)
24 VD=TRANS(GS,BY,YMAX)
25 WRITE(1,*) UD,VD,' lineto stroke'
26 C ****
27 C * SIDE LABELS. *
28 C ****
29 WRITE(1,*) '/Times-Roman findfont 10 scalefont setfont'
30 XE=XMIN
31 YE=YMAX
32 UD=TRANS(GS,BX,XE)
33 VD=TRANS(GS,BY,YE)
34 WRITE(1,*) UD,VD+6.0,' moveto'
35 WRITE(1,*) '(side 1) show'
36 XE=XMIN+A/2
37 YE=YMIN
38 UD=TRANS(GS,BX,XE)
39 VD=TRANS(GS,BY,YE)
40 WRITE(1,*) UD,VD,' moveto'
41 WRITE(1,*) '/Times-Italic findfont 10 scalefont setfont'
42 WRITE(1,*) '(y = b/2) stringwidth pop neg 2 div -10 rmoveto'
43 WRITE(1,*) '(y = b) show'
44 WRITE(1,*) '/Times-Roman findfont 10 scalefont setfont'
45 WRITE(1,*) '(/2) show'
46 XE=XMIN+A/2.0
47 YE=YMAX
48 UD=TRANS(GS,BX,XE)
49 VD=TRANS(GS,BY,YE)
50 WRITE(1,*) '0.4 setlinewidth'
51 AO=10.0
52 WRITE(1,*) UD,VD+AO,' moveto'
53 WRITE(1,*) UD,VD+AO+15.0,' lineto stroke'
54 WRITE(1,*) UD-3,VD+AO+15.0-5.0,' moveto'
55 WRITE(1,*) UD,VD+AO+15.0,' lineto'
56 WRITE(1,*) UD+3,VD+AO+15.0-5.0,' lineto stroke'
57 WRITE(1,*) UD-11.0,VD+15.0,' moveto'
58 WRITE(1,*) '/Times-Italic findfont 10 scalefont setfont'
59 WRITE(1,*) '(z) show'
60 XE=-B
61 YE=YMAX
62 UD=TRANS(GS,BX,XE)
63 VD=TRANS(GS,BY,YE)
64 WRITE(1,*) UD,VD+6.0,' moveto'
65 WRITE(1,*) '/Times-Roman findfont 10 scalefont setfont'

```

```

66      WRITE(1,*) '(side 2) show'
67      XE=XMIN+A+B/2
68      YE=YMIN
69      UD=TRANS(GS,BX,XE)
70      VD=TRANS(GS,BY,YE)
71      WRITE(1,*) UD,VD,' moveto'
72      WRITE(1,*) '/Times-Italic findfont 10 scalefont setfont'
73      WRITE(1,*) '(x = a/2) stringwidth pop neg 2 div -10 rmoveto'
74      WRITE(1,*) '(x = a) show'
75      WRITE(1,*) '/Times-Roman findfont 10 scalefont setfont'
76      WRITE(1,*) '(/2) show'
77      XE=0.0
78      YE=YMAX
79      UD=TRANS(GS,BX,XE)
80      VD=TRANS(GS,BY,YE)
81      WRITE(1,*) UD,VD+6.0,' moveto'
82      WRITE(1,*) '/Times-Roman findfont 10 scalefont setfont'
83      WRITE(1,*) '(side 3) show'
84      XE=XMIN+A+B+A/2
85      YE=YMIN
86      UD=TRANS(GS,BX,XE)
87      VD=TRANS(GS,BY,YE)
88      WRITE(1,*) UD,VD,' moveto'
89      WRITE(1,*) '/Times-Italic findfont 10 scalefont setfont'
90      WRITE(1,*) '(y = -b/2) stringwidth pop neg 2 div -10 rmoveto'
91      WRITE(1,*) '(y = -b) show'
92      WRITE(1,*) '/Times-Roman findfont 10 scalefont setfont'
93      WRITE(1,*) '(/2) show'
94      XE=A
95      YE=YMAX
96      UD=TRANS(GS,BX,XE)
97      VD=TRANS(GS,BY,YE)
98      WRITE(1,*) UD,VD+6.0,' moveto'
99      WRITE(1,*) '/Times-Roman findfont 10 scalefont setfont'
100     WRITE(1,*) '(side 4) show'
101     XE=XMAX-B/2
102     YE=YMIN
103     UD=TRANS(GS,BX,XE)
104     VD=TRANS(GS,BY,YE)
105     WRITE(1,*) UD,VD,' moveto'
106     WRITE(1,*) '/Times-Italic findfont 10 scalefont setfont'
107     WRITE(1,*) '(x = -a/2) stringwidth pop neg 2 div -10 rmoveto'
108     WRITE(1,*) '(x = -a) show'
109     WRITE(1,*) '/Times-Roman findfont 10 scalefont setfont'
110     WRITE(1,*) '(/2) show'
111 C ****
112 C * Z-AXIS LABELS. *
113 C ****
114     TL=7.0
115     PL=10.0
116     KT=KM+1
117     XE=XMAX
118     UD=TRANS(GS,BX,XE)
119     DY=(YMAX-YMIN)/(KT-1)
120     DO 1 I=0,KT-1
121       YE=YMIN+I*DY
122       VD=TRANS(GS,BY,YE)
123       WRITE(1,*) UD,VD,' moveto'
124       WRITE(1,*) UD+TL,VD,' lineto stroke'

```

```

125   1  CONTINUE
126   VD=TRANS(GS,BY,YMIN)
127   WRITE(1,*) UD+TL+PL,VD,' moveto gsave'
128   WRITE(1,*) UD+TL+PL,VD,' translate 90 rotate'
129   WRITE(1,*) '/Times-Roman findfont 10 scalefont setfont'
130   WRITE(1,*) '(0) stringwidth pop 2 div neg 0 rmoveto'
131   WRITE(1,*) '(0) show'
132   WRITE(1,*) 'grestore'
133   VD=TRANS(GS,BY,YMIN+DY)
134   WRITE(1,*) UD+TL+PL,VD,' moveto gsave'
135   WRITE(1,*) UD+TL+PL,VD,' translate 90 rotate'
136   WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
137   WRITE(1,*) '(\160 \244 \142) stringwidth pop 2 div neg 0 rmoveto'
138   WRITE(1,*) '(\160 \244 \142) show'
139   WRITE(1,*) 'grestore'
140   VD=TRANS(GS,BY,YMIN+2*DY)
141   WRITE(1,*) UD+TL+PL,VD,' moveto gsave'
142   WRITE(1,*) UD+TL+PL,VD,' translate 90 rotate'
143   WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
144   WRITE(1,*) '(2\160 \244 \142) stringwidth pop 2 div neg 0 rmoveto'
145   WRITE(1,*) '(2\160 \244 \142) show'
146   WRITE(1,*) 'grestore'
147   VD=TRANS(GS,BY,YMIN+3*DY)
148   WRITE(1,*) UD+TL+PL,VD,' moveto gsave'
149   WRITE(1,*) UD+TL+PL,VD,' translate 90 rotate'
150   WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
151   WRITE(1,*) '(3\160 \244 \142) stringwidth pop 2 div neg 0 rmoveto'
152   WRITE(1,*) '(3\160 \244 \142) show'
153   WRITE(1,*) 'grestore'
154   VD=TRANS(GS,BY,YMIN+4*DY)
155   WRITE(1,*) UD+TL+PL,VD,' moveto gsave'
156   WRITE(1,*) UD+TL+PL,VD,' translate 90 rotate'
157   WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
158   WRITE(1,*) '(4\160 \244 \142) stringwidth pop 2 div neg 0 rmoveto'
159   WRITE(1,*) '(4\160 \244 \142) show'
160   WRITE(1,*) 'grestore'
161   VD=TRANS(GS,BY,YMIN+5*DY)
162   WRITE(1,*) UD+TL+PL,VD,' moveto gsave'
163   WRITE(1,*) UD+TL+PL,VD,' translate 90 rotate'
164   WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
165   WRITE(1,*) '(5\160 \244 \142) stringwidth pop 2 div neg 0 rmoveto'
166   WRITE(1,*) '(5\160 \244 \142) show'
167   WRITE(1,*) 'grestore'
168   RETURN
169   END

1  SUBROUTINE CLABEL(XMIN,XMAX,YMIN,YMAX,GS,BX,BY,A,KM)
2 C ****
3 C * THIS SUBROUTINE DRAWS THE LABELS FOR THE CIRCULAR GUIDE. *
4 C ****
5 C ****
6 C * Z-AXIS LABELS. *
7 C ****
8 XE=XMIN
9 YE=YMAX
10 UD=TRANS(GS,BX,XE)
11 VD=TRANS(GS,BY,YE)
12 WRITE(1,*) '0.4 setlinewidth'
13 AO=10.0

```

```

14      WRITE(1,*) UD,VD+AO,' moveto'
15      WRITE(1,*) UD,VD+AO+15.0,' lineto stroke'
16      WRITE(1,*) UD-3,VD+AO+15.0-5.0,' moveto'
17      WRITE(1,*) UD,VD+AO+15.0,' lineto'
18      WRITE(1,*) UD+3,VD+AO+15.0-5.0,' lineto stroke'
19      WRITE(1,*) UD+11.0,VD+15.0,' moveto'
20      WRITE(1,*) '/Times-Italic findfont 10 scalefont setfont'
21      WRITE(1,*) '(z) show'
22      TL=7.0
23      PL=10.0
24      KT=KM+1
25      XE=XMAX
26      UD=TRANS(GS,BX,XE)
27      DY=(YMAX-YMIN)/(KT-1)
28      DO 1 I=0,KT-1
29          YE=YMIN+I*DY
30          VD=TRANS(GS,BY,YE)
31          WRITE(1,*) UD,VD,' moveto'
32          WRITE(1,*) UD+TL,VD,' lineto stroke'
33 1    CONTINUE
34      VD=TRANS(GS,BY,YMIN)
35      WRITE(1,*) UD+TL+PL,VD,' moveto gsave'
36      WRITE(1,*) UD+TL+PL,VD,' translate 90 rotate'
37      WRITE(1,*) '/Times-Roman findfont 10 scalefont setfont'
38      WRITE(1,*) '(0) stringwidth pop 2 div neg 0 rmoveto'
39      WRITE(1,*) '(0) show'
40      WRITE(1,*) 'grestore'
41      VD=TRANS(GS,BY,YMIN+DY)
42      WRITE(1,*) UD+TL+PL,VD,' moveto gsave'
43      WRITE(1,*) UD+TL+PL,VD,' translate 90 rotate'
44      WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
45      WRITE(1,*) '(\160 \244 \142) stringwidth pop 2 div neg 0 rmoveto'
46      WRITE(1,*) '(\160 \244 \142) show'
47      WRITE(1,*) 'grestore'
48      VD=TRANS(GS,BY,YMIN+2*DY)
49      WRITE(1,*) UD+TL+PL,VD,' moveto gsave'
50      WRITE(1,*) UD+TL+PL,VD,' translate 90 rotate'
51      WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
52      WRITE(1,*) '(2\160 \244 \142) stringwidth pop 2 div neg 0 rmoveto'
53      WRITE(1,*) '(2\160 \244 \142) show'
54      WRITE(1,*) 'grestore'
55      VD=TRANS(GS,BY,YMIN+3*DY)
56      WRITE(1,*) UD+TL+PL,VD,' moveto gsave'
57      WRITE(1,*) UD+TL+PL,VD,' translate 90 rotate'
58      WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
59      WRITE(1,*) '(3\160 \244 \142) stringwidth pop 2 div neg 0 rmoveto'
60      WRITE(1,*) '(3\160 \244 \142) show'
61      WRITE(1,*) 'grestore'
62      VD=TRANS(GS,BY,YMIN+4*DY)
63      WRITE(1,*) UD+TL+PL,VD,' moveto gsave'
64      WRITE(1,*) UD+TL+PL,VD,' translate 90 rotate'
65      WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
66      WRITE(1,*) '(4\160 \244 \142) stringwidth pop 2 div neg 0 rmoveto'
67      WRITE(1,*) '(4\160 \244 \142) show'
68      WRITE(1,*) 'grestore'
69      VD=TRANS(GS,BY,YMIN+5*DY)
70      WRITE(1,*) UD+TL+PL,VD,' moveto gsave'
71      WRITE(1,*) UD+TL+PL,VD,' translate 90 rotate'
72      WRITE(1,*) '/Symbol findfont 10 scalefont setfont'

```

```

73      WRITE(1,*) '(5\160 \244 \142) stringwidth pop 2 div neg 0 rmoveto'
74      WRITE(1,*) '(5\160 \244 \142) show'
75      WRITE(1,*) 'grestore'
76 C ****
77 C * PHI-AXIS LABELS. *
78 C ****
79      YE=YMIN
80      VD=TRANS(GS,BY,YE)
81      NPT=5
82      DX=(XMAX-XMIN)/(NPT-1)
83      DO 2 I=0,NPT-1
84          XE=XMIN+I*DX
85          UD=TRANS(GS,BX,XE)
86          WRITE(1,*) UD,VD,' moveto'
87          WRITE(1,*) UD,VD-TL,' lineto stroke'
88 2    CONTINUE
89      UD=TRANS(GS,BX,XMIN)
90      WRITE(1,*) UD,VD-TL-PL,' moveto'
91      WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
92      WRITE(1,*) '(0) stringwidth pop 2 div neg 0 rmoveto'
93      WRITE(1,*) '(0) show'
94      WRITE(1,*) '(\260) show'
95      UD=TRANS(GS,BX,XMIN+DX)
96      WRITE(1,*) UD,VD-TL-PL,' moveto'
97      WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
98      WRITE(1,*) '(90) stringwidth pop 2 div neg 0 rmoveto'
99      WRITE(1,*) '(90) show'
100     WRITE(1,*) '(\260) show'
101     UD=TRANS(GS,BX,XMIN+2*DX)
102     WRITE(1,*) UD,VD-TL-PL,' moveto'
103     WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
104     WRITE(1,*) '(180) stringwidth pop 2 div neg 0 rmoveto'
105     WRITE(1,*) '(180) show'
106     WRITE(1,*) '(\260) show'
107     UD=TRANS(GS,BX,XMIN+3*DX)
108     WRITE(1,*) UD,VD-TL-PL,' moveto'
109     WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
110     WRITE(1,*) '(270) stringwidth pop 2 div neg 0 rmoveto'
111     WRITE(1,*) '(270) show'
112     WRITE(1,*) '(\260) show'
113     UD=TRANS(GS,BX,XMIN+4*DX)
114     WRITE(1,*) UD,VD-TL-PL,' moveto'
115     WRITE(1,*) '/Symbol findfont 10 scalefont setfont'
116     WRITE(1,*) '(360) stringwidth pop 2 div neg 0 rmoveto'
117     WRITE(1,*) '(360) show'
118     WRITE(1,*) '(\260) show'
119     RETURN
120 END

1      SUBROUTINE RWTE10(MN,MT,MP,NT,N2T,T2N,X,Y,A,B,KM,M,N,XMIN,XMAX
2      & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
3 C ****
4 C * THIS SUBROUTINE DRAWS VECTOR LINES FOR A RECTANGULAR *
5 C * WAVEGUIDE WITH TE10 MODE FIELDS. *
6 C ****
7 C * INPUTS: *
8 C *
9 C *   MN - MAXIMUM NUMBER OF COORDINATE POINTS. *
10 C *  MT - MAXIMUM NUMBER OF TRIANGLES. *

```

```

11 C      * N2T(MT,3) - NODE MATRIX WHERE FIRST INDEX REPRESENTS      *
12 C      * EACH TRIANGLE AND SECOND INDEX REPRESENTS      *
13 C      * THE NODE NUMBERS.      *
14 C      * T2N(MN,MP) - MATRIX OF TRIANGLE TO NODE CONNECTIONS      *
15 C      * WHERE THE FIRST INDEX IS THE NODE NUMBER      *
16 C      * AND THE SECOND IS THE LOCAL TRIANGLE NUMBER.      *
17 C      * X(MN) - COORDINATE VECTORS OF GRID.      *
18 C      * Y(MN)      *
19 C      * FX(MN) - VECTOR FUNCTION FIELD VALUES AT EACH COORDINATE.      *
20 C      * FY(MN)      *
21 C      * AX,BX - GRAPHICAL SCALE CONSTANTS.      *
22 C      * AY,BY      *
23 C      * VL - SPACING BETWEEN ARROWHEADS IN REAL COORDINATES.      *
24 C      * BANG - ARROW APEX HALF ANGLE IN DEGREES.      *
25 C      * S - ARROW HEAD SIDE LENGTH IN POINT SIZE.      *
26 C      * MNAR - MAXIMUM NUMBER OF ARROWS ALLOWED.      *
27 C      * DS - STEP SIZE.      *
28 C      *      *
29 C      * OUTPUTS:      *
30 C      *      *
31 C ****
32 REAL*4 X(MN),Y(MN),FX(MN),FY(MN),KC2
33 INTEGER N2T(MT,3),T2N(MN,MP)
34 PI=.3141593E+01
35 RAD=.17453293E-01
36 KC2=(M*PI/A)*(M*PI/A)+(N*PI/B)*(N*PI/B)
37 BETA=SQRT(4*PI*PI-KC2)
38 C ****
39 C      * SET GLOBAL VECTOR INPUTS.      *
40 C ****
41 AH=S*COS(RAD*BANG)
42 C ****
43 C      * SET SPACING PARAMETERS BASED ON CRITICAL POINT LOCATIONS.      *
44 C ****
45 TAUY=0.7
46 NPY=5
47 NPX=9
48 TAUX=0.96
49 DL=PI/BETA
50 DX=TAUX*A/(NPX-1)
51 DY=(TAUY*DL)/(NPY-1)
52 C ****
53 C      * SIDE 1 (Y=B/2).      *
54 C ****
55 C ****
56 C      * DRAW VECTOR LINES ALONG A VERTICAL LINE WHICH ORIGINATE ON      *
57 C      * THE PERIMETER OF SIDE 1 AND TERMINATE AT CRITICAL POINT 1.      *
58 C ****
59 K=0
60 DO 1 I=1,(NPY-1)/2
61 K=K+1
62 WRITE(*,*) 'DRAWING VECTOR ',K
63 YE=I*DY
64 XE=-B
65 CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
66 NDIR=0
67 VL=20.0/GS
68 CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
69 ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)

```

```

70      NDIR=0
71      XE=-B-A
72      VL=20.0/GS
73      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
74      CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
75      & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
76 1    CONTINUE
77 C ****
78 C * DRAW VECTOR LINES ALONG A HORIZONTAL LINE WHICH ORIGINATE ON *
79 C * CRITICAL POINT 2 AND TERMINATE ON CRITICAL POINT 1. *
80 C ****
81      DXOFF=(1.0-TAUX)*A/2
82      DO 2 I=0,KM-1
83          YE=0.5*DL+I*DL
84          DO 3 J=0,NPX-1
85              K=K+1
86              WRITE(*,*) 'DRAWING VECTOR ',K
87              XE=-B-A+DXOFF+J*DX
88              CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
89              NDIR=0
90              VL=27.0/GS
91              CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
92      & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
93              NDIR=1
94              VL=27.0/GS-AH/GS
95              CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
96      & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
97 3    CONTINUE
98 2    CONTINUE
99 C ****
100 C * DRAW VECTOR LINES ALONG A VERTICAL LINE WHICH ORIGINATE AT *
101 C * CRITICAL POINT 2 AND TERMINATE ON THE PERIMETER OF SIDE 1. *
102 C ****
103      DO 4 I=0,KM-2
104          YCE=(1.0-TAUY)*0.5*DL+0.5*DL+I*DL
105          IS=(-1)**I
106          IF (IS > 0) THEN
107              NDIR=1
108              VL=20.0/GS-S/GS
109          ELSE
110              NDIR=0
111              VL=20.0/GS
112          END IF
113          DO 5 J=0,NPY-1
114              K=K+1
115              WRITE(*,*) 'DRAWING VECTOR ',K
116              YE=YCE+J*DY
117              XE=-B
118              CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
119              CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
120      & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
121              XE=-B-A
122              CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
123              CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
124      & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
125 5    CONTINUE
126 4    CONTINUE
127 C ****
128 C * DRAW VECTOR LINES ALONG A VERTICAL LINE WHICH ORIGINATE AT *

```

```

129 C * CRITICAL POINT 5 AND TERMINATE ON THE PERIMETER OF SIDE 1. *
130 C ****
131 NP=(NPY-1)/2
132 DO 6 I=0,NP-1
133 K=K+1
134 WRITE(*,*) 'DRAWING VECTOR ',K
135 YE=4.5*DL+(1.0-TAUY)*0.5*DL+I*DY
136 XE=-B
137 CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
138 NDIR=1
139 VL=27.0/GS-AH/GS
140 CALL VEC2D(XE,YE, ID, NDIR, MN, MT, MP, NT, N2T, T2N, X, Y, XMIN, XMAX
141 & ,YMIN, YMAX, FX, FY, GS, BX, BY, VL, BANG, S, MNAR, DS)
142 NDIR=1
143 VL=27.0/GS-AH/GS
144 XE=-B-A
145 CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
146 CALL VEC2D(XE,YE, ID, NDIR, MN, MT, MP, NT, N2T, T2N, X, Y, XMIN, XMAX
147 & ,YMIN, YMAX, FX, FY, GS, BX, BY, VL, BANG, S, MNAR, DS)
148 6 CONTINUE
149 C ****
150 C * SIDE 3 (Y=-B/2). *
151 C ****
152 C ****
153 C * DRAW VECTOR LINES ALONG A VERTICAL LINE WHICH ORIGINATE ON *
154 C * THE PERIMETER OF SIDE 1 AND TERMINATE AT CRITICAL POINT 1. *
155 C ****
156 DO 7 I=1,(NPY-1)/2
157 K=K+1
158 WRITE(*,*) 'DRAWING VECTOR ',K
159 YE=I*DY
160 XE=0
161 CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
162 NDIR=1
163 VL=20.0/GS-AH/GS
164 CALL VEC2D(XE,YE, ID, NDIR, MN, MT, MP, NT, N2T, T2N, X, Y, XMIN, XMAX
165 & ,YMIN, YMAX, FX, FY, GS, BX, BY, VL, BANG, S, MNAR, DS)
166 NDIR=1
167 XE=A
168 VL=20.0/GS-AH/GS
169 CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
170 CALL VEC2D(XE,YE, ID, NDIR, MN, MT, MP, NT, N2T, T2N, X, Y, XMIN, XMAX
171 & ,YMIN, YMAX, FX, FY, GS, BX, BY, VL, BANG, S, MNAR, DS)
172 7 CONTINUE
173 C ****
174 C * DRAW VECTOR LINES ALONG A HORIZONTAL LINE WHICH ORIGINATE ON *
175 C * CRITICAL POINT 2 AND TERMINATE ON CRITICAL POINT 1. *
176 C ****
177 DXOFF=(1.0-TAUX)*A/2
178 DO 8 I=0,KM-1
179 YE=0.5*DL+I*DL
180 DO 9 J=0,NPX-1
181 K=K+1
182 WRITE(*,*) 'DRAWING VECTOR ',K
183 XE=DXOFF+J*DX
184 CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
185 NDIR=0
186 VL=27.0/GS
187 CALL VEC2D(XE,YE, ID, NDIR, MN, MT, MP, NT, N2T, T2N, X, Y, XMIN, XMAX

```

```

188      &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
189          NDIR=1
190          VL=27.0/GS-AH/GS
191          CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
192      &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
193      9    CONTINUE
194      8    CONTINUE
195 C   ****
196 C   * DRAW VECTOR LINES ALONG A VERTICAL LINE WHICH ORIGINATE AT      *
197 C   * CRITICAL POINT 2 AND TERMINATE ON THE PERIMETER OF SIDE 1.      *
198 C   ****
199 DO 10 I=0,KM-2
200     YCE=(1.0-TAUY)*0.5*DL+0.5*DL+I*DL
201     IS=(-1)**(I+1)
202     IF (IS > 0) THEN
203         NDIR=1
204         VL=20.0/GS-S/GS
205     ELSE
206         NDIR=0
207         VL=20.0/GS
208     END IF
209     DO 11 J=0,NPY-1
210         K=K+1
211         WRITE(*,*) 'DRAWING VECTOR ',K
212         YE=YCE+J*DY
213         XE=0
214         CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
215         CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
216      &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
217         XE=A
218         CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
219         CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
220      &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
221      11    CONTINUE
222      10    CONTINUE
223 C   ****
224 C   * DRAW VECTOR LINES ALONG A VERTICAL LINE WHICH ORIGINATE AT      *
225 C   * CRITICAL POINT 5 AND TERMINATE ON THE PERIMETER OF SIDE 1.      *
226 C   ****
227     NP=(NPY-1)/2
228     DO 12 I=0,NP-1
229         K=K+1
230         WRITE(*,*) 'DRAWING VECTOR ',K
231         YE=4.5*DL+(1.0-TAUY)*0.5*DL+I*DY
232         XE=0
233         CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
234         NDIR=0
235         VL=27.0/GS
236         CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
237      &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
238         NDIR=0
239         VL=27.0/GS
240         XE=A
241         CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
242         CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
243      &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
244      12    CONTINUE
245 C   ****
246 C   * SIDE 2 (X=A/2).      *

```

```

247 C ****
248 C ****
249 C * END POINT CONTRIBUTIONS. *
250 C ****
251 XMIN=-B
252 XMAX=0.0
253 XE=(XMIN+XMAX)/2.0
254 NP=(NPY-1)/2
255 DO 13 I=0,NP-1
256     YE=4.5*DL+(1.0-TAUY)*0.5*DL+I*DY
257     K=K+1
258     WRITE(*,*) 'DRAWING VECTOR ',K
259     CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
260     NDIR=0
261     VL=20.0/GS
262     CALL VEC2D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
263     & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
264     NDIR=1
265     VL=20.0/GS-S/GS
266     CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
267     CALL VEC2D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
268     & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
269     YE=(I+1)*DY
270     K=K+1
271     WRITE(*,*) 'DRAWING VECTOR ',K
272     CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
273     NDIR=0
274     VL=20.0/GS
275     CALL VEC2D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
276     & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
277     NDIR=1
278     VL=20.0/GS-S/GS
279     CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
280     CALL VEC2D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
281     & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
282
283 13 CONTINUE
284 C ****
285 C * INTERIOR CONTRIBUTIONS. *
286 C ****
287 DO 14 I=0,KM-2
288     YCE=(1.0-TAUY)*0.5*DL+0.5*DL+I*DL
289     DO 15 J=0,NPY-1
290         K=K+1
291         WRITE(*,*) 'DRAWING VECTOR ',K
292         YE=YCE+J*DY
293         NDIR=1
294         VL=20.0/GS-S/GS
295         CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
296         CALL VEC2D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
297         & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
298         NDIR=0
299         VL=20.0/GS
300         CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
301         CALL VEC2D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
302         & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
303 15 CONTINUE
304 14 CONTINUE
305 C ****

```

```

306 C      * SIDE 4 (X==A/2). *
307 C      ****
308 C      ****
309 C      * END POINT CONTRIBUTIONS. *
310 C      ****
311 XMIN=A
312 XMAX=A+B
313 XE=(XMIN+XMAX)/2.0
314 NP=(NPY-1)/2
315 DO 16 I=0,NP-1
316     YE=4.5*DL+(1.0-TAUY)*^.5*DL+I*DY
317     K=K+1
318     WRITE(*,*) 'DRAWING VECTOR ',K
319     CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
320     NDIR=0
321     VL=20.0/GS
322     CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
323     &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
324     NDIR=1
325     VL=20.0/GS-S/GS
326     CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
327     CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
328     &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
329     YE=(I+1)*DY
330     K=K+1
331     WRITE(*,*) 'DRAWING VECTOR ',K
332     CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
333     NDIR=0
334     VL=20.0/GS
335     CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
336     &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
337     NDIR=1
338     VL=20.0/GS-S/GS
339     CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
340     CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
341     &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
342
343 16  CONTINUE
344 C      ****
345 C      * INTERIOR CONTRIBUTIONS. *
346 C      ****
347 DO 17 I=0,KM-2
348     YCE=(1.0-TAUY)*0.5*DL+0.5*DL+I*DL
349     DO 18 J=0,NPY-1
350         K=K+1
351         WRITE(*,*) 'DRAWING VECTOR ',K
352         YE=YCE+J*DY
353         NDIR=1
354         VL=20.0/GS-S/GS
355         CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
356         CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
357         &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
358         NDIR=0
359         VL=20.0/GS
360         CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
361         CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
362         &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
363 18  CONTINUE
364 17  CONTINUE

```

```

365      RETURN
366      END

1      SUBROUTINE CWTE10(MN,MT,MP,NT,N2T,T2N,X,Y,A,C,KM,N,M,XMIN,XMAX
2      &                               ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
3 C      ****
4 C      * THIS SUBROUTINE DRAWS VECTOR LINES FOR A CIRCULAR          *
5 C      * WAVEGUIDE WITH TE11 MODE FIELDS.                         *
6 C      ****
7 C      * INPUTS:                                         *
8 C      *
9 C      *   XE, YE - FIRST COORDINATE OF VECTOR LINE.           *
10 C      *   MN  - MAXIMUM NUMBER OF COORDINATE POINTS.        *
11 C      *   MT  - MAXIMUM NUMBER OF TRIANGLES.                  *
12 C      *   N2T(MT,3) - NODE MATRIX WHERE FIRST INDEX REPRESENTS   *
13 C      *                   EACH TRIANGLE AND SECOND INDEX REPRESENTS   *
14 C      *                   THE NODE NUMBERS.                      *
15 C      *   T2N(MN,MP) - MATRIX OF TRIANGLE TO NODE CONNECTIONS   *
16 C      *                   WHERE THE FIRST INDEX IS THE NODE NUMBER   *
17 C      *                   AND THE SECOND IS THE LOCAL TRIANGLE NUMBER. *
18 C      *   X(MN) - COORDINATE VECTORS OF GRID.                 *
19 C      *   Y(MN)                                         *
20 C      *   FX(MN) - VECTOR FUNCTION FIELD VALUES AT EACH COORDINATE. *
21 C      *   FY(MN)                                         *
22 C      *   AX,BX - GRAPHICAL SCALE CONSTANTS.                *
23 C      *   AY,BY                                         *
24 C      *   VL   - SPACING BETWEEN ARROWHEADS IN REAL COORDINATES. *
25 C      *   BANG - ARROW APEX HALF ANGLE IN DEGREES.          *
26 C      *   S    - ARROW HEAD SIDE LENGTH IN POINT SIZE.       *
27 C      *   MNAR - MAXIMUM NUMBER OF ARROWS ALLOWED.          *
28 C      *   DS   - STEP SIZE.                                *
29 C      *
30 C      * OUTPUTS:                                         *
31 C      *
32 C      ****
33      REAL*4 X(MN),Y(MN),FX(MN),FY(MN)
34      INTEGER N2T(MT,3),T2N(MN,MP)
35      PI=.3141593E+01
36      TP=.6283185E+01
37      RAD=.17453293E-01
38      K=0
39 C      ****
40 C      * SET GLOBAL VECTOR INPUTS.                         *
41 C      ****
42      AH=S*COS(RAD*BANG)
43 C      ****
44 C      * SET SPACING PARAMETERS BASED ON CRITICAL POINT LOCATIONS. *
45 C      ****
46      TAUY=0.9
47      NPY=5
48      DY=(TAUY*C/KM)/(NPY-1)
49 C      ****
50 C      * LEFT REGION HORIZONTAL LINES.                    *
51 C      ****
52      XE=0.0
53      DO 1 I=0,KM
54          YMIN=-C/(2*KM)+I*C/KM
55          YMAX=YMIN+C/KM
56          DO 2 J=0,NPY-1

```

```

57      YE=YMIN+(C/KM)*(1.0-TAUY)/2.0+J*DY
58      IF ((YE .GT. DY/2.0) .AND. (YE .LT. C-DY/2.0)) THEN
59          K=K+1
60          WRITE(*,*) 'DRAWING VECTOR ',K
61          CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
62          NDIR=0
63          VL=20.0/GS
64          CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
65          & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
66          NDIR=1
67          VL=20.0/GS-S/GS
68          CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
69          & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
70          ELSE
71          END IF
72 2      CONTINUE
73 1      CONTINUE
74 C      ****
75 C      * LEFT REGION VERTICAL LINES. *
76 C      ****
77 XWIDTH=PI*A
78 TAUX=0.90
79 NPX=9
80 DX=TAUX*XWIDTH/(NPX-1)
81 DO 3 I=0,KM-1
82     YE=C/(2*KM)+I*C/KM
83     YMIN=YE-C/(2*KM)
84     YMAX=YE+C/(2*KM)
85     DO 4 J=0,NPX-1
86         K=K+1
87         WRITE(*,*) 'DRAWING VECTOR ',K
88         XE=XWIDTH*(1.0-TAUX)/2.0+J*DX
89         CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
90         NDIR=0
91         VL=20.0/GS
92         CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
93         & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
94         NDIR=1
95         VL=20.0/GS-S/GS
96         CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
97         & ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
98 4      CONTINUE
99 3      CONTINUE
100 C      ****
101 C      * CENTER REGION HORIZONTAL LINES. *
102 C      ****
103 XE=PI*A
104 DO 5 I=0,KM
105     YMIN=-C/(2*KM)+I*C/KM
106     YMAX=YMIN+C/KM
107     DO 6 J=0,NPY-1
108         YE=YMIN+(C/KM)*(1.0-TAUY)/2.0+J*DY
109         IF ((YE .GT. DY/2.0) .AND. (YE .LT. C-DY/2.0)) THEN
110             K=K+1
111             WRITE(*,*) 'DRAWING VECTOR ',K
112             CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
113             NDIR=0
114             VL=20.0/GS
115             CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX

```

```

116      &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
117          NDIR=1
118          VL=20.0/GS-S/GS
119          CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
120      &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
121          ELSE
122          END IF
123      6    CONTINUE
124      5    CONTINUE
125 C   ****
126 C   * RIGHT REGION VERTICAL LINES. *
127 C   ****
128 XWIDTH=PI*A
129 TAUX=0.90
130 DX=TAUX*XWIDTH/(NPX-1)
131 DO 7 I=0,KM-1
132     YE=C/(2*KM)+I*C/KM
133     YMIN=YE-C/(2*KM)
134     YMAX=YE+C/(2*KM)
135     DO 8 J=0,NPX-1
136         K=K+1
137         WRITE(*,*) 'DRAWING VECTOR ',K
138         XE=PI*A+XWIDTH*(1.0-TAUX)/2.0+J*DX
139         CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,IDL)
140         NDIR=0
141         VL=20.0/GS
142         CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
143             &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
144         NDIR=1
145         VL=20.0/GS-S/GS
146         CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
147             &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
148      8    CONTINUE
149      7    CONTINUE
150 C   ****
151 C   * RIGHT REGION HORIZONTAL LINES. *
152 C   ****
153 XE=TP*A
154 DO 9 I=0,KM
155     YMIN=-C/(2*KM)+I*C/KM
156     YMAX=YMIN+C/KM
157     DO 10 J=0,NPY-1
158         YE=YMIN+(C/KM)*(1.0-TAUY)/2.0+J*DY
159         IF ((YE .GT. DY/2.0) .AND. (YE .LT. C-DY/2.0)) THEN
160             K=K+1
161             WRITE(*,*) 'DRAWING VECTOR ',K
162             CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,IDL)
163             NDIR=0
164             VL=20.0/GS
165             CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
166                 &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
167             NDIR=1
168             VL=20.0/GS-S/GS
169             CALL VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
170                 &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
171             ELSE
172             END IF
173      10   CONTINUE
174      9    CONTINUE

```

```

175      RETURN
176      END

1      SUBROUTINE VEC2D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
2      &           ,YMIN,YMAX,FX,FY,GS,BX,BY,VL,BANG,S,MNAR,DS)
3 C      ****
4 C      * THIS SUBROUTINE DRAWS A VECTOR IN TWO DIMENSIONS. *
5 C      ****
6 C      * INPUTS: *
7 C      *
8 C      * XE,YE - FIRST COORDINATE OF VECTOR LINE. *
9 C      * MN - MAXIMUM NUMBER OF COORDINATE POINTS. *
10 C      * MT - MAXIMUM NUMBER OF TRIANGLES. *
11 C      * N2T(MT,3) - NODE MATRIX WHERE FIRST INDEX REPRESENTS *
12 C      * EACH TRIANGLE AND SECOND INDEX REPRESENTS *
13 C      * THE NODE NUMBERS. *
14 C      * T2N(MN,MP) - MATRIX OF TRIANGLE TO NODE CONNECTIONS *
15 C      * WHERE THE FIRST INDEX IS THE NODE NUMBER *
16 C      * AND THE SECOND IS THE LOCAL TRIANGLE NUMBER. *
17 C      * X(MN) - COORDINATE VECTORS OF GRID. *
18 C      * Y(MN) *
19 C      * FX(MN) - VECTOR FUNCTION FIELD VALUES AT EACH COORDINATE. *
20 C      * FY(MN) *
21 C      * AX,BX - GRAPHICAL SCALE CONSTANTS. *
22 C      * AY,BY *
23 C      * VL - SPACING BETWEEN ARROWHEADS IN REAL COORDINATES. *
24 C      * BANG - ARROW APEX HALF ANGLE IN DEGREES. *
25 C      * S - ARROW HEAD SIDE LENGTH IN POINT SIZE. *
26 C      * MNAR - MAXIMUM NUMBER OF ARROWS ALLOWED. *
27 C      * DS - STEP SIZE. *
28 C      *
29 C      * OUTPUTS: *
30 C      *
31 C      ****
32      REAL*4 X(MN),Y(MN),FX(MN),FY(MN)
33      INTEGER N2T(MT,3),T2N(MN,MP)
34      MXSTEP=300
35 C      ****
36 C      * STORE THE FIRST TRIANGLE NUMBER. *
37 C      ****
38      IFIRST=ID
39 C      ****
40 C      * STORE THE FIRST POINT. *
41 C      ****
42      XFIRST=XE
43      YFIRST=YE
44      XAT=TRANS(GS,BX,XE)
45      YAT=TRANS(GS,BY,YE)
46      WRITE(1,100) XAT,YAT,' moveto'
47 C      ****
48 C      * BEGIN THE VECTOR SEGMENTS. *
49 C      ****
50      K=0
51      VSUM=0.0
52      KAR=0
53      XA=XE
54      YA=YE
55 999      CONTINUE
56      K=K+1

```

```

57 C      WRITE(*,*) 'SEGMENT NUMBER= ',K,' TRIANGLE NUMBER= ',ID
58 C ****
59 C * IF THIS IS NOT THE FIRST LINE THEN DETERMINE WHAT TRIANGLE *
60 C * THE DRAWN LINE CONNECTS TO BY LOOKING AT THE TWO NODES NA AND *
61 C * NB FORMING THE LAST INTERSECTION. *
62 C ****
63 IF (K .GT. 1) THEN
64   IF (K .GT. MXSTEP) THEN
65     WRITE(*,*) 'MAX STEPS EXCEEDED '
66     GO TO 99
67   ELSE
68   END IF
69   XA=XF
70   YA=YF
71   IF (ID .EQ. 0) THEN
72 C ****
73 C * THERE IS NO NEXT TRIANGLE WHICH MEANS THE VECTOR *
74 C * TERMINATES ON A BOUNDARY. *
75 C ****
76   WRITE(*,*) 'TERMINATE ON BOUNDARY TRIANGLE'
77   GO TO 99
78 ELSE IF (ID .EQ. IFIRST) THEN
79 C   WRITE(*,*) 'BACK TO ORIGINAL TRIANGLE'
80 C   GO TO 99
81   ELSE
82   END IF
83 ELSE
84 END IF
85 C ****
86 C * FIND THE VECTOR FUNCTION VALUE AT THE POINT (XA,YA). *
87 C ****
88 CALL VECFCOF(MN,MT,ID,X,Y,XA,YA,N2T,FX,FY,A,B,E,C,D,F)
89 C ****
90 C * GET THE VECTOR COMPONENTS AT THE POINT (XA,YA). *
91 C ****
92 FYA=A*XA+B*YA+E
93 FXA=C*XA+D*YA+F
94 FM=SQRT(FXA*FXA+FYA*FYA)
95 FXN=FXA/FM
96 FYN=FYA/FM
97 C ****
98 C * INCREMENT OR DECREMENT THE VECTOR FROM THE POINT (XA,YA). *
99 C ****
100 IF (NDIR .EQ. 0) THEN
101 C ****
102 C * INCREMENT. *
103 C ****
104 XF=XA+DS*FXN
105 YF=YA+DS*FYN
106 ELSE
107 C ****
108 C * DECREMENT. *
109 C ****
110 XF=XA-DS*FXN
111 YF=YA-DS*FYN
112 END IF
113 C ****
114 C * GET THE NEXT TRIANGLE NUMBER. *
115 C ****

```

```

116      IDOLD=ID
117      CALL FDSTRI(XF,YF,MN,X,Y,MT,MP,NT,N2T,T2N,IDL0D,ID)
118      IF (ID .EQ. IFIRST) THEN
119 C          XF=XFIRST
120 C          YF=YFIRST
121      ELSE IF (ID .EQ. 0) THEN
122 C          WRITE(*,*) 'BOUNDARY NODE'
123 C          ****
124 C          * ADJUST THE POINT IF IT IS OUTSIDE THE BOUNDARY. *
125 C          ****
126      IF (XF .LT. XMIN) THEN
127          XF=XMIN
128      ELSE IF (XF .GT. XMAX) THEN
129          XF=XMAX
130      ELSE
131      END IF
132      IF (YF .LT. YMIN) THEN
133          YF=YMIN
134      ELSE IF (YF .GT. YMAX) THEN
135          YF=YMAX
136      ELSE
137      END IF
138 C          ****
139 C          * DRAW THE LINE FROM (XA,YA) TO (XF,YF). *
140 C          ****
141      XFT=TRANS(GS,BX,XF)
142      YFT=TRANS(GS,BY,YF)
143      WRITE(1,100) XFT,YFT,' lineto'
144      GO TO 99
145  END IF
146 C          ****
147 C          * ADJUST THE POINT IF IT IS OUTSIDE THE BOUNDARY. *
148 C          ****
149      IF (XF .LT. XMIN) THEN
150          XF=XMIN
151      ELSE IF (XF .GT. XMAX) THEN
152          XF=XMAX
153      ELSE
154      END IF
155      IF (YF .LT. YMIN) THEN
156          YF=YMIN
157      ELSE IF (YF .GT. YMAX) THEN
158          YF=YMAX
159      ELSE
160      END IF
161 C          ****
162 C          * DRAW THE LINE FROM (XA,YA) TO (XF,YF). *
163 C          ****
164      XFT=TRANS(GS,BX,XF)
165      YFT=TRANS(GS,BY,YF)
166      WRITE(1,100) XFT,YFT,' lineto'
167 C          ****
168 C          * DRAW ARROW OR ARROWS IF APPROPRIATE. *
169 C          ****
170      DAR=SQRT((XA-XF)*(XA-XF)+(YA-YF)*(YA-YF))
171      IF (DAR .GT. 0.0) THEN
172          IF (NDIR .EQ. 0) THEN
173              CALL ARR2D(MNAR,VL,BANG,S,XA,YA,XF,YF,FXN,FYN,VSUM,KAR
174              & ,GS,BX,BY)

```

```

175      ELSE
176          CALL ARRW2D(MNAR,VL,BANG,S,XF,YF,XA,YA,FXN,FYN,VSUM,KAR
177                      ,GS,BX,BY)
178      &      END IF
179      ELSE
180      END IF
181 C      ****
182 C      * STOP IF VECTOR MAGNITUDE DROPS BELOW TOLERANCE. *
183 C      ****
184 IF (FM .LT. .001) THEN
185 C      WRITE(*,*) 'CLOSE TO AN UNKNOWN CRITICAL POINT'
186      GO TO 99
187      ELSE
188      END IF
189      GO TO 999
190 C      ****
191 C      * STCP. *
192 C      ****
193 99      CONTINUE
194      WRITE(1,*) 'stroke'
195 100     FORMAT(2(F6.2,1X),A7)
196      ID=IFIRST
197      RETURN
198      END

1      SUBROUTINE ARRW2D(MNAR,VL,BANG,S,XA,YA,XF,YF,FXN,FYN,VSUM,KAR
2      &                      ,GS,BX,BY)
3 C      ****
4 C      * THIS SUBROUTINE CONTROLS THE DRAWING OF ARROWHEADS ON A LINE. *
5 C      ****
6 C      * INPUTS: *
7 C      *
8 C      * MNAR   - MAXIMUM NUMBER OF ARROWS ALLOWED. *
9 C      * VL     - SPACING BETWEEN ARROWHEADS IN REAL COORDINATES. *
10 C      * BANG    - ARROW APEX HALF ANGLE IN DEGREES. *
11 C      * S       - ARROW HEAD SIDE LENGTH IN POINT SIZE. *
12 C      * XA, YA - BASE OF ARROW. *
13 C      * XF, YF - TIP OF ARROW. *
14 C      * FXN, FYN - NORMALIZED VECTOR FUNCTION VALUES. *
15 C      * VSUM    - ACCUMULATED LINE LENGTH. *
16 C      * KAR     - NUMBER OF ARROWS DRAWN SO FAR. *
17 C      * AX, BX - GRAPHICAL SCALE CONSTANTS. *
18 C      * AY, BY -
19 C      ****
20      D=SQRT((XA-XF)*(XA-XF)+(YA-YF)*(YA-YF))
21      STEP=VSUM+D
22      IF (STEP .GT. VL) THEN
23          KAR=KAR+1
24          IF (KAR .GT. MNAR) THEN
25              GO TO 2
26          ELSE
27          END IF
28      ****
29 C      * THE NEXT LINE SEGMENT IS LONGER THAN THE ARROW SPACING. *
30 C      * SO AN ARROW SHOULD BE DRAWN. *
31 C      ****
32      RAT=(VL-VSUM)/D
33      XFW=XA+RAT*(XF-XA)
34      YFW=YA+RAT*(YF-YA)

```

```

35 C ****
36 C * DRAW THE ARROW. *
37 C ****
38 XFJ=TRANS(GS,BX,XFW)
39 YFJ=TRANS(GS,BY,YFW)
40 CALL AHED2D(BANG,S,XFJ,YFJ,FXN,FYN)
41 C ****
42 C * COMPUTE THE AMOUNT LEFT OVER AFTER THE ARROW IS DRAWN. *
43 C ****
44 DLT=SQRT((XF-XFW)*(XF-XFW)+(YF-YFW)*(YF-YFW))
45 IF (DLT .LT. VL) THEN
46 C ****
47 C * NO MORE ARROWS ARE DRAWN ON THIS SEGMENT. *
48 C ****
49 VSUM=DLT
50 ELSE
51 C ****
52 C * MORE THAN ONE ARROW IS DRAWN ON THIS SEGMENT. *
53 C ****
54 NAR=DLT/VL
55 DO 1 I=1,NAR
56 KAR=KAR+1
57 IF (KAR .GT. MNAR) THEN
58 GO TO 2
59 ELSE
60 END IF
61 RAT=(VL-VSUM+I*VL)/D
62 XFW=XA+RAT*(XF-XA)
63 YFW=YA+RAT*(YF-YA)
64 XFJ=TRANS(GS,BX,XFW)
65 YFJ=TRANS(GS,BY,YFW)
66 CALL AHED2D(BANG,S,XFJ,YFJ,FXN,FYN)
67 1 CONTINUE
68 VSUM=SQRT((XF-XFW)*(XF-XFW)+(YF-YFW)*(YF-YFW))
69 END IF
70 ELSE
71 C ****
72 C * THE LINE LENGTH PLUS THE ACCUMULATED DISTANCE IS STILL LESS *
73 C * THAN THE ARROW SPACING. NO ARROW SHOULD BE DRAWN JUST *
74 C * ACCUMULATE THE DISTANCE. *
75 C ****
76 VSUM=VSUM+D
77 END IF
78 2 CONTINUE
79 RETURN
80 END

1 SUBROUTINE AHED2D(BANG,S,X2,Y2,FXN,FYN)
2 C ****
3 C * THIS SUBROUTINE DRAWS A ARROW FROM (X1,Y1) TO (X2,Y2). *
4 C ****
5 C * INPUTS: *
6 C *
7 C * BANG - ARROW APEX HALF ANGLE IN DEGREES. *
8 C * S - ARROWHEAD SIDE LENGTH. *
9 C * X2,Y2 - TIP OF ARROW. *
10 C * FXN,FYN - NORMALIZED VECTOR FUNCTION VALUES. *
11 C ****
12 RAD=.17453293E-01

```

```

13      X1=X2-FXN
14      Y1=Y2-FYN
15      D=SQRT((X2-X1)*(X2-X1)+(Y2-Y1)*(Y2-Y1))
16      A=1.0-S/D
17      X3=X1+A*(X2-X1)
18      Y3=Y1+A*(Y2-Y1)
19 C      ****
20 C      * TRANSLATE THE ORIGIN TO (X2,Y2) AND ROTATE (X3,Y3) PLUS AND *
21 C      * MINUS BANG DEGREES TO GENERATE THE VERTICES OF THE ARROW.   *
22 C      ****
23      X0=X2
24      Y0=Y2
25      XA=X3-X2
26      YA=Y3-Y2
27      ARG=RAD*BANG
28      CA=COS(ARG)
29      SA=SIN(ARG)
30      XCA=XA*CA
31      XSA=XA*SA
32      YCA=YA*CA
33      YSA=YA*SA
34      X4=X0+XCA-YSA
35      Y4=Y0+XSA+YCA
36      X5=X0+XCA+YSA
37      Y5=Y0-XSA+YCA
38 C      X6=(X4+X5)/2.0
39 C      Y6=(Y4+Y5)/2.0
40 C      WRITE(1,*) X1,Y1,' moveto'
41 C      WRITE(1,*) X6,Y6,' lineto stroke'
42 C      ****
43 C      * TRANSLATE THE ORIGIN TO (X2,Y2) AND ROTATE (X3,Y3) PLUS AND *
44 C      * MINUS BANG DEGREES TO GENERATE THE VERTICES OF THE ARROW   *
45 C      ****
46      WRITE(1,*) 'gsave newpath'
47      WRITE(1,*) X2,Y2,' moveto ',X4,Y4,' lineto ',X5,Y5,' lineto'
48      WRITE(1,*) 'closepath 0 setgray fill stroke grestore'
49      RETURN
50      END

```

```

1      PROGRAM FLDREC
2 C
3 C      * THIS PROGRAM GENERATES A VECTOR PLOT OF THE TRIANGULARIZED   *
4 C      * REGION OF A RECTANGULAR WAVEGUIDE.                           *
5 C
6 C      * TIMOTHY J. PETERS          LAST UPDATED   *
7 C      * THE AEROSPACE CORPORATION      5/12/92       *
8 C      * 2350 EAST EL SEGUNDO BOULEVARD      *
9 C      * EL SEGUNDO, CA 90245           *
10 C
11 C      PARAMETER (MN=6000,MT=12000,MP=6,NC=5,NL=6)
12 C      REAL*4 U(MN),V(MN),FU(MN),FV(MN),PHI(50),CA(NC),CB(NC)
13 C      REAL*4 XQ(3),YQ(3),ZQ(3)
14 C      REAL*8 ARG
15 C      INTEGER LC(NC),ITMIN(4),ITMAX(4),ISIDE(4)
16 C      CHARACTER*30 LABEL(NL)
17 C      INTEGER*4 N2T(MT,3),T2N(MN,MP)
18 C      PI=.3141593E+01
19 C      RAD=.17453293E-01
20 C
21 C      * INPUTS:                                *
22 C      *
23 C      * GS - GRAPH SCALE CONSTANT.          *
24 C      * PO - PHI OBSERVATION ANGLE (DEGREES).    *
25 C      * TO - THETA OBSERVATION ANGLE (DEGREES).  *
26 C
27 C      GS=180.0
28 C      PO=60.0
29 C      TO=60.0
30 C
31 C      * READ THE COORDINATE DATA AND VECTOR FUNCTION DATA FROM A FILE. *
32 C
33 C      OPEN(UNIT=1,FILE='RUVFDF')
34 C      READ(1,*) A,B,C,KM,M,N
35 C      READ(1,*) NUA,NUB,NV
36 C      READ(1,*) NN
37 C      VMAX=0.0
38 C      DO 1 I=1,NN
39 C          READ(1,*) U(I),V(I),FU(I),FV(I)
40 C          VMAG=SQRT(FU(I)*FU(I)+FV(I)*FV(I))
41 C          IF (VMAG .GT. VMAX) THEN
42 C              VMAX=VMAG
43 C          ELSE
44 C              END IF
45 C      1 CONTINUE
46 C      CLOSE(1)
47 C      WRITE(*,*) NN,' DATA POINTS READ IN'
48 C
49 C      * NORMALIZE THE VECTOR COMPONENTS SO THAT THE MAXIMUM MAGNITUDE   *
50 C      * IS 1.0.                                         *
51 C
52 C      DO 2 I=1,NN
53 C          FU(I)=FU(I)/VMAX
54 C          FV(I)=FV(I)/VMAX
55 C      2 CONTINUE
56 C
57 C      * READ THE TRIANGLE NODE DATA.                         *
58 C
59 C      OPEN(UNIT=1,FILE='N2TDAT')

```

```

60      READ(1,*) NT
61      DO 3 I=1,NT
62          READ(1,*) N2T(I,1),N2T(I,2),N2T(I,3)
63 3    CONTINUE
64      CLOSE(1)
65      WRITE(*,*) 'NODE TO TRIANGLE DATA READ IN'
66 C   ****
67 C   * READ THE TRIANGLE TO NODE CONNECTION DATA. *
68 C   ****
69      OPEN(UNIT=1,FILE='T2NDAT')
70      READ(1,*) NN
71      DO 4 I=1,NN
72          READ(1,*) (T2N(I,J),J=1,MP)
73 4    CONTINUE
74      CLOSE(1)
75      WRITE(*,*) 'TRIANGLE TO NODE DATA READ IN'
76 C   ****
77 C   * COMPUTE CENTROID OF GRAPH. *
78 C   ****
79      XMIN=-B/2.0
80      XMAX=B/2.0
81      YMIN=-A/2.0
82      YMAX=A/2.0
83      ZMIN=0.0
84      ZMAX=C
85      XC=(XMIN+XMAX)/2.0
86      YC=(YMIN+YMAX)/2.0
87      ZC=(ZMIN+ZMAX)/2.0
88 C   ****
89 C   * CONVERT THE OBSERVATION ANGLES TO RADIANS. *
90 C   ****
91      RAD=.17453293E-01
92      PI=.3141593E+01
93      PI4=PI/4.0
94      XD=XC
95      YD=YC
96      ZD=ZC
97      CALL SCALE(GS,XD,YD,ZD)
98 C   ****
99 C   * CONVERT THE OBSERVATION ANGLES TO RADIANS. *
100 C   ****
101      POBS=RAD*PO
102      TOBS=RAD*TO
103 C   ****
104 C   * COMPUTE THE ROTATION ANGLES WHICH YIELD THE DESIRED OBSERVATION*
105 C   * ANGLES. *
106 C   ****
107      RP=-POBS-PI/2.0
108      RT=TOBS
109 C   ****
110 C   * COMPUTE THE CONSTANTS FOR THE ROTATION MATRIX. *
111 C   ****
112      CALL RCOEFF(RP,RT,C1,C2,C3,C4,C5)
113 C   ****
114 C   * COMPUTE THE CENTER OF THE PRINTING DEVICE PAGE. *
115 C   ****
116      X0=72*8.5/2.0
117      Y0=72*11.0/2.0
118 C   ****

```

```

119 C * SET THE GRAPHIC BOUNDING BOX. *
120 C ****
121 WX=(XMAX-XMIN)
122 WY=(YMAX-YMIN)
123 WIDTH=GS*WX
124 HEIGHT=GS*WY
125 UA=X0-WIDTH/2.0-60.0
126 UB=X0+WIDTH/2.0+100.0
127 VA=Y0-HEIGHT/2.0-180.0
128 VB=Y0+HEIGHT/2.0+180.0
129 C ****
130 C * COMPUTE THE BOUNDING BOX FOR THE POSTSCRIPT IN THE DEFAULT *
131 C * COORDINATE SYSTEM. *
132 C ****
133 OPEN(UNIT=1,FILE='FoldedRect.ps')
134 WRITE(1,*) '%!PS-Adobe-1.0'
135 WRITE(1,*) '%%%Creator: Timothy J. Peters'
136 WRITE(1,*) '%%%Title: Graph'
137 WRITE(1,*) '%%%CreationDate: 5-14-92'
138 WRITE(1,100) '%%%BoundingBox:',INT(UA),INT(VA),INT(UB),INT(VB)
139 100 FORMAT(A14,4(1X,I3))
140 WRITE(1,*) '%%%EndComments'
141 WRITE(1,*) '/dot2 {2 0 360 arc 0 setgray fill stroke} def'
142 WRITE(1,*) '/dot3 {3 0 360 arc 0 setgray fill stroke} def'
143 WRITE(1,*) '/dot4 {4 0 360 arc 0 setgray fill stroke} def'
144 WRITE(1,*) '/black {0 0 0 setrgbcolor} def'
145 WRITE(1,*) '/white {1 1 1 setrgbcolor} def'
146 WRITE(1,*) '/gray-lt {0.92 0.92 0.92 setrgbcolor} def'
147 WRITE(1,*) '/gray-lt-med {0.65 0.65 0.65 setrgbcolor} def'
148 WRITE(1,*) '/gray {0.45 0.45 0.45 setrgbcolor} def'
149 WRITE(1,*) '/gray-dk-med {0.3 0.3 0.3 setrgbcolor} def'
150 WRITE(1,*) '/gray-dk {0.14 0.14 0.14 setrgbcolor} def'
151 WRITE(1,*) '/red {1 0 0 setrgbcolor} def'
152 WRITE(1,*) '/magenta {1 0 1 setrgbcolor} def'
153 WRITE(1,*) '/green {0 1 0 setrgbcolor} def'
154 WRITE(1,*) '/blue {0 0 1 setrgbcolor} def'
155 WRITE(1,*) '/cyan {0 1 1 setrgbcolor} def'
156 WRITE(1,*) '/yellow {1 1 0 setrgbcolor} def'
157 WRITE(1,*) '/orange {1 0.5 0 setrgbcolor} def'
158 WRITE(1,*) '/brown {0.5 0.5 0 setrgbcolor} def'
159 WRITE(1,*) '/kakhi {0.5 1 0 setrgbcolor} def'
160 WRITE(1,*) '/blue-lt {0.5 1 1 setrgbcolor} def'
161 WRITE(1,*) '/green-lt {0.5 1 0.5 setrgbcolor} def'
162 WRITE(1,*) '/green-blue {0 1 0.5 setrgbcolor} def'
163 WRITE(1,*) '/purple {0.6 0 1.0 setrgbcolor} def'
164 WRITE(1,*) '/filtri {moveto lineto lineto'
165 WRITE(1,*) ' closepath fill stroke} def'
166 WRITE(1,*) '/filqud {moveto lineto lineto lineto closepath fill'
167 WRITE(1,*) ' stroke} def'
168 WRITE(1,*) '/filpnt {moveto lineto lineto lineto'
169 WRITE(1,*) ' lineto closepath fill stroke} def'
170 WRITE(1,*) '%%%EndProlog'
171 WRITE(1,*) '1 setlinejoin .4 setlinewidth'
172 C ****
173 C * DRAW THE EPS BOUNDING BOX. *
174 C ****
175 C WRITE(1,*) 'gsave 0.2 setlinewidth'
176 C WRITE(1,*) UA,VA,' moveto'
177 C WRITE(1,*) UB,VA,' lineto'

```

```

178 C      WRITE(1,*) UB,VB,' lineto'
179 C      WRITE(1,*) UA,VB,' lineto'
180 C      WRITE(1,*) 'closepath stroke grestore'
181 C ****
182 C * ASSIGN CONTOUR VALUES AND LABELS. *
183 C ****
184     CA(1)=0.0
185     CB(1)=0.15
186     CA(2)=0.15
187     CB(2)=0.4
188     CA(3)=0.4
189     CB(3)=0.6
190     CA(4)=0.6
191     CB(4)=0.8
192     CA(5)=0.8
193     CB(5)=1.0
194     LABEL(1)='0'
195     LABEL(2)='15'
196     LABEL(3)='4'
197     LABEL(4)='6'
198     LABEL(5)='8'
199     LABEL(6)='1'
200 C ****
201 C * ASSIGN THE COLORS. SEE SUBROUTINE SETCOL FOR COLOR CHOICES. *
202 C ****
203 C GRAY SCALE
204     LC(1)=2
205     LC(2)=3
206     LC(3)=4
207     LC(4)=5
208     LC(5)=6
209 C COLOR
210 C     LC(1)=18
211 C     LC(2)=14
212 C     LC(3)=12
213 C     LC(4)=16
214 C     LC(5)=13
215 C ****
216 C * CALCULATE AND STORE   *
217 C ****
218     N1MIN=1
219     N1MAX=N1MIN+2*(NUA-1)*(NV-1)-1
220     N2MIN=N1MAX+1
221     N2MAX=N2MIN+2*(NUB-1)*(NV-1)-1
222     N3MIN=N2MAX+1
223     N3MAX=N3MIN+2*(NUA-1)*(NV-1)-1
224     N4MIN=N3MAX+1
225     N4MAX=N4MIN+2*(NUB-1)*(NV-1)-1
226 IF ((PO .GE. 0.0) .AND. (PO .LE. 90.0)) THEN
227 C ****
228 C * DRAW SIDES IN ORDER 3,4,2,1.   *
229 C ****
230     ISIDE(1)=3
231     ISIDE(2)=4
232     ISIDE(3)=2
233     ISIDE(4)=1
234     ITMIN(1)=N3MIN
235     ITMAX(1)=N3MAX
236     ITMIN(2)=N4MIN

```

```

237      ITMAX(2)=N4MAX
238      ITMIN(3)=N2MIN
239      ITMAX(3)=N2MAX
240      ITMIN(4)=N1MIN
241      ITMAX(4)=N1MAX
242      ELSE
243      END IF
244 C      **** STEP THROUGH EACH SIDE FROM BACK TO FRONT. ****
245 C      * DRAW THE CONTOUR PLOT FOR THIS SIDE. *
246 C      ****
247 DO 6 J=1,4
248      WRITE(*,*) 'SIDE ',J
249 C      ****
250 C      * DRAW THE CONTOUR PLOT FOR THIS SIDE. *
251 C      ****
252 DO 7 I=ITMIN(J),ITMAX(J)
253      N1=N2T(I,1)
254      N2=N2T(I,2)
255      N3=N2T(I,3)
256      F1=SQRT(FU(N1)*FU(N1)+FV(N1)*FV(N1))
257      F2=SQRT(FU(N2)*FU(N2)+FV(N2)*FV(N2))
258      F3=SQRT(FU(N3)*FU(N3)+FV(N3)*FV(N3))
259      CALL MPRPNT(U(N1),V(N1),A,B,XQ(1),YQ(1),ZQ(1))
260      CALL MPRPNT(U(N2),V(N2),A,B,XQ(2),YQ(2),ZQ(2))
261      CALL MPRPNT(U(N3),V(N3),A,B,XQ(3),YQ(3),ZQ(3))
262 C      ****
263 C      * TRANSFORM THE 3 VERTICES OF THE TRIANGLE. *
264 C      ****
265      CALL TRFTRI(XQ,YQ,ZQ,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
266 C      ****
267 C      * DRAW THE CONTOUR. *
268 C      ****
269      CALL CONTR(XQ(1),YQ(1),XQ(2),YQ(2),XQ(3),YQ(3),F1,F2,F3
270      ,CA,CB,NC,LC)
271      7 CONTINUE
272 C      ****
273 C      * DRAW THE PERIMETER AROUND THE SIDE. *
274 C      ****
275 IF (ISIDE(J) .EQ. 1) THEN
276      U1=-A-B
277      V1=0.0
278      U2=-B
279      V2=0.0
280      U3=-B
281      V3=C
282      U4=-A-B
283      V4=C
284 ELSE IF (ISIDE(J) .EQ. 2) THEN
285      U1=-B
286      V1=0.0
287      U2=0.0
288      V2=0.0
289      U3=0.0
290      V3=C
291      U4=-B
292      V4=C
293 ELSE IF (ISIDE(J) .EQ. 3) THEN
294      U1=0.0
295      V1=0.0

```

```

296      U2=A
297      V2=0.0
298      U3=A
299      V3=C
300      U4=0.0
301      V4=C
302      ELSE
303          U1=A
304          V1=0.0
305          U2=A+B
306          V2=0.0
307          U3=A+B
308          V3=C
309          U4=A
310          V4=C
311      END IF
312      CALL MPPRN(T(U1,V1,A,B,X1,Y1,Z1)
313      CALL MPPRN(T(U2,V2,A,B,X2,Y2,Z2)
314      CALL MPPRN(T(U3,V3,A,B,X3,Y3,Z3)
315      CALL MPPRN(T(U4,V4,A,B,X4,Y4,24)
316      CALL TRFPNT(X1,Y1,Z1,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
317      CALL TRFPNT(X2,Y2,Z2,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
318      CALL TRFPNT(X3,Y3,Z3,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
319      CALL TRFPNT(X4,Y4,Z4,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
320      WRITE(1,*) 'gsave 0 setgray 0.4 setlinewidth newpath'
321      WRITE(1,*) X1,Y1,' moveto ',X2,Y2,' lineto'
322      WRITE(1,*) X3,Y3,' lineto ',X4,Y4,' lineto'
323      WRITE(1,*) 'closepath stroke grestore'
324 C ****
325 C * DRAW THE VECTORS FOR THIS SIDE. *
326 C ****
327      BANG=20.0
328      S=8.0
329      MNAR=1
330      DS=.003
331      UMIN=-A-B
332      UMAX=A+B
333      VMIN=0.0
334      VMAX=5.0/SQRT(3.0)
335      WRITE(1,*) '0 setgray'
336      CALL RWT3D(NDIR,MN,MT,MP,NT,N2T,T2N,U,V,A,B,KM,UMIN,UMAX
337      & ,VMIN,VMAX,FU,FV,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
338      & ,S,MNAR,DS,PO,ISIDE(J))
339      6 CONTINUE
340 C ****
341 C * DRAW THE LEGEND. *
342 C ****
343      SL=15.0
344      SH=30.0
345      DS=0.0
346      ITYPE=1
347      US=420.0
348      VS=173.0
349      CALL LEGEND(US,VS,NC,NL,LC,LABEL,SL,SH,DS,ITYPE)
350 C ****
351 C * DRAW THE PAGE. *
352 C ****
353      WRITE(1,*) 'showpage'
354      CLOSE(1)

```

```

355      END

1      SUBROUTINE RWTE3D(NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,A,B,KM,XMIN,XMAX
2      &,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
3      &,S,MNAR,DS,PO,IS)
4 C      ****
5 C      * THIS SUBROUTINE DRAWS VECTOR LINES FOR THE TE10 MODE FIELDS. *
6 C      ****
7 C      * INPUTS: *
8 C      *
9 C      * MN - MAXIMUM NUMBER OF COORDINATE POINTS. *
10 C      * MT - MAXIMUM NUMBER OF TRIANGLES. *
11 C      * N2T(MT,3) - NODE MATRIX WHERE FIRST INDEX REPRESENTS *
12 C      * EACH TRIANGLE AND SECOND INDEX REPRESENTS *
13 C      * THE NODE NUMBERS. *
14 C      * T2N(MN,MP) - MATRIX OF TRIANGLE TO NODE CONNECTIONS *
15 C      * WHERE THE FIRST INDEX IS THE NODE NUMBER *
16 C      * AND THE SECOND IS THE LOCAL TRIANGLE NUMBER. *
17 C      * X(MN) - COORDINATE VECTORS OF GRID. *
18 C      * Y(MN) *
19 C      * FX(MN) - VECTOR FUNCTION FIELD VALUES AT EACH COORDINATE. *
20 C      * FY(MN) *
21 C      * AX,BX - GRAPHICAL SCALE CONSTANTS. *
22 C      * AY,BY *
23 C      * VL - SPACING BETWEEN ARROWHEADS IN REAL COORDINATES. *
24 C      * BANG - ARROW APEX HALF ANGLE IN DEGREES. *
25 C      * S - ARROW HEAD SIDE LENGTH IN POINT SIZE. *
26 C      * MNAR - MAXIMUM NUMBER OF ARROWS ALLOWED. *
27 C      * DS - STEP SIZE. *
28 C      *
29 C      * OUTPUTS: *
30 C      *
31 C      ****
32      REAL*4 X(MN),Y(MN),FX(MN),FY(MN)
33      INTEGER N2T(MT,3),T2N(MN,MP)
34      PI=.3141593E+01
35      RAD=.17453293E-01
36      AH=S*COS(RAD*BANG)
37 C      ****
38 C      * SET SPACING PARAMETERS BASED ON CRITICAL POINT LOCATIONS. *
39 C      ****
40      NPY=5
41      NPX=9
42      TAUX=0.96
43      TAUY=0.7
44      DL=1.0/SQRT(3.0)
45      DX=TAUX*A/(NPX-1)
46      DY=(TAUY*DL)/(NPY-1)
47      IF (IS .EQ. 1) THEN
48 C      ****
49 C      * DRAW SIDE 1 (Y=B/2) VECTORS. *
50 C      ****
51      XMIN=-A-B
52      XMAX=-B
53 C      ****
54 C      * DRAW VECTOR LINES ALONG A VERTICAL LINE WHICH ORIGINATE ON *
55 C      * THE PERIMETER OF SIDE 1 AND TERMINATE AT CRITICAL POINT 1. *
56 C      ****
57      K=0

```

```

58      DO 1 I=1,(NPY-1)/2
59          K=K+1
60          WRITE(*,*) 'DRAWING VECTOR ',K
61          YE=I*DY
62          XE=-B
63          CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
64          NDIR=0
65          VL=20.0/GS
66          CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
67          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
68          & ,S,MNAR,DS)
69          NDIR=0
70          XE=-B-A
71          VL=20.0/GS
72          CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
73          CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
74          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
75          & ,S,MNAR,DS)
76 1    CONTINUE
77 C   ****
78 C   * DRAW VECTOR LINES ALONG A HORIZONTAL LINE WHICH ORIGINATE ON *
79 C   * CRITICAL POINT 2 AND TERMINATE ON CRITICAL POINT 1.           *
80 C   ****
81 DXOFF=(1.0-TAUX)*A/2
82 DO 2 I=0,KM-1
83     YE=0.5*DL+I*DL
84     DO 3 J=0,NPX-1
85         K=K+1
86         WRITE(*,*) 'DRAWING VECTOR ',K
87         XE=-B-A+DXOFF+J*DX
88         CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
89         NDIR=0
90         VL=27.0/GS
91         CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
92         & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
93         & ,S,MNAR,DS)
94         NDIR=1
95         VL=27.0/GS-AH/GS
96         CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
97         & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
98         & ,S,MNAR,DS)
99 3    CONTINUE
100 2   CONTINUE
101 C   ****
102 C   * DRAW VECTOR LINES ALONG A VERTICAL LINE WHICH ORIGINATE AT   *
103 C   * CRITICAL POINT 2 AND TERMINATE ON THE PERIMETER OF SIDE 1.       *
104 C   ****
105 DO 4 I=0,KM-2
106     YCE=(1.0-TAUY)*0.5*DL+0.5*DL+I*DL
107     IS=(-1)**I
108     IF (IS > 0) THEN
109         NDIR=1
110         VL=20.0/GS-S/GS
111     ELSE
112         NDIR=0
113         VL=20.0/GS
114     END IF
115     DO 5 J=0,NPY-1
116         K=K+1

```

```

117      WRITE(*,*) 'DRAWING VECTOR ',K
118      YE=YCE+J*DY
119      XE=-B
120      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
121      CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
122      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
123      & ,S,MNAR,DS)
124      XE=-B-A
125      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
126      CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
127      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
128      & ,S,MNAR,DS)
129      5  CONTINUE
130      4  CONTINUE
131 C  *****
132 C  * DRAW VECTOR LINES ALONG A VERTICAL LINE WHICH ORIGINATE AT   *
133 C  * CRITICAL POINT 5 AND TERMINATE ON THE PERIMETER OF SIDE 1.   *
134 C  *****
135      NP=(NPY-1)/2
136      DO 6 I=0,NP-1
137          K=K+1
138          WRITE(*,*) 'DRAWING VECTOR ',K
139          YE=4.5*DL+(1.0-TAUY)*0.5*DL+I*DY
140          XE=-B
141          CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
142          NDIR=1
143          VL=27.0/GS-AH/GS
144          CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
145          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
146          & ,S,MNAR,DS)
147          NDIR=1
148          VL=27.0/GS-AH/GS
149          XE=-B-A
150          CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
151          CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
152          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
153          & ,S,MNAR,DS)
154      6  CONTINUE
155      ELSE IF (IS .EQ. 2) THEN
156 C  *****
157 C  * DRAW SIDL 2 VECTORS (X=A/2).           *
158 C  *****
159 C  *****
160 C  * END POINT CONTRIBUTIONS.             *
161 C  *****
162      XMIN=-B
163      XMAX=0.0
164      XE=(XMIN+XMAX)/2.0
165      NP=(NPY-1)/2
166      K=0
167      DO 7 I=0,NP-1
168          YE=4.5*DL+(1.0-TAUY)*0.5*DL+I*DY
169          K=K+1
170          WRITE(*,*) 'DRAWING VECTOR ',K
171          CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
172          NDIR=0
173          VL=20.0/GS
174          CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
175          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG

```

```

176      & ,S,MNAR,DS)
177      NDIR=1
178      VL=20.0/GS-S/GS
179      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
180      CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
181      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
182      & ,S,MNAR,DS)
183      YE=(I+1)*DY
184      K=K+1
185      WRITE(*,*) 'DRAWING VECTOR ',K
186      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
187      NDIR=0
188      VL=20.0/GS
189      CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
190      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
191      & ,S,MNAR,DS)
192      NDIR=1
193      VL=20.0/GS-S/GS
194      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
195      CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
196      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
197      & ,S,MNAR,DS)
198      7 CONTINUE
199 C ****
200 C * INTERIOR CONTRIBUTIONS. *
201 C ****
202 DO 8 I=0,KM-2
203     YCE=(1.0-TAUY)*0.5*DL+0.5*DL+I*DL
204     DO 9 J=0,NPY-1
205         K=K+1
206         WRITE(*,*) 'DRAWING VECTOR ',K
207         YE=YCE+J*DY
208         NDIR=1
209         VL=20.0/GS-S/GS
210         CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
211         CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
212         & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
213         & ,S,MNAR,DS)
214         NDIR=0
215         VL=20.0/GS
216         CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
217         CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
218         & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
219         & ,S,MNAR,DS)
220      9 CONTINUE
221      8 CONTINUE
222 ELSE IF (IS .EQ. 3) THEN
223 C ****
224 C * DRAW SIDE 3 (Y=-B/2) VECTORS. *
225 C ****
226 C ****
227 C * DRAW VECTOR LINES ALONG A VERTICAL LINE WHICH ORIGINATE ON *
228 C * THE PERIMETER OF SIDE 1 AND TERMINATE AT CRITICAL POINT 1. *
229 C ****
230 K=0
231 DO 10 I=1,(NPY-1)/2
232     K=K+1
233     WRITE(*,*) 'DRAWING VECTOR ',K
234     YE=I*DY

```

```

235      XE=0
236      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
237      NDIR=1
238      VL=20.0/GS-AH/GS
239      CALL RVEC3D(XE,YE, ID, NDIR, MN, MT, MP, NT, N2T, T2N, X, Y, XMIN, XMAX
240      & , YMIN, YMAX, FX, FY, GS, XD, YD, ZD, C1, C2, C3, C4, C5, X0, Y0, VL, BANG
241      & , S, MNAR, DS)
242      NDIR=1
243      XE=A
244      VL=20.0/GS-AH/GS
245      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
246      CALL RVEC3D(XE,YE, ID, NDIR, MN, MT, MP, NT, N2T, T2N, X, Y, XMIN, XMAX
247      & , YMIN, YMAX, FX, FY, GS, XD, YD, ZD, C1, C2, C3, C4, C5, X0, Y0, VL, BANG
248      & , S, MNAR, DS)
249      10  CONTINUE
250      C ****
251      C * DRAW VECTOR LINES ALONG A HORIZONTAL LINE WHICH ORIGINATE ON *
252      C * CRITICAL POINT 2 AND TERMINATE ON CRITICAL POINT 1. *
253      C ****
254      DXOFF=(1.0-TAUX)*A/2
255      DO 11 I=0,KM-1
256          YE=0.5*DL+I*DL
257          DO 12 J=0,NPX-1
258              K=K+1
259              WRITE(*,*) 'DRAWING VECTOR ',K
260              XE=DXOFF+J*DX
261              CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
262              NDIR=0
263              VL=27.0/GS
264              CALL RVEC3D(XE,YE, ID, NDIR, MN, MT, MP, NT, N2T, T2N, X, Y, XMIN, XMAX
265              & , YMIN, YMAX, FX, FY, GS, XD, YD, ZD, C1, C2, C3, C4, C5, X0, Y0, VL, BANG
266              & , S, MNAR, DS)
267              NDIR=1
268              VL=27.0/GS-AH/GS
269              CALL RVEC3D(XE,YE, ID, NDIR, MN, MT, MP, NT, N2T, T2N, X, Y, XMIN, XMAX
270              & , YMIN, YMAX, FX, FY, GS, XD, YD, ZD, C1, C2, C3, C4, C5, X0, Y0, VL, BANG
271              & , S, MNAR, DS)
272      12  CONTINUE
273      11  CONTINUE
274      C ****
275      C * DRAW VECTOR LINES ALONG A VERTICAL LINE WHICH ORIGINATE AT *
276      C * CRITICAL POINT 2 AND TERMINATE ON THE PERIMETER OF SIDE 1. *
277      C ****
278      DO 13 I=0,KM-2
279          YCE=(1.0-TAUY)*0.5*DL+0.5*DL+I*DL
280          IS=(-1)**(I+1)
281          IF (IS > 0) THEN
282              NDIR=1
283              VL=20.0/GS-S/GS
284          ELSE
285              NDIR=0
286              VL=20.0/GS
287          END IF
288          DO 14 J=0,NPY-1
289              K=K+1
290              WRITE(*,*) 'DRAWING VECTOR ',K
291              YE=YCE+J*DY
292              XE=0
293              CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)

```

```

294      CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
295      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
296      & ,S,MNAR,DS)
297      XE=A
298      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
299      CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
300      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
301      & ,S,MNAR,DS)
302 14    CONTINUE
303 13    CONTINUE
304 C   ****
305 C   * DRAW VECTOR LINES ALONG A VERTICAL LINE WHICH ORIGINATE AT *
306 C   * CRITICAL POINT 5 AND TERMINATE ON THE PERIMETER OF SIDE 1. *
307 C   ****
308      NP=(NPY-1)/2
309      DO 15 I=0,NP-1
310      K=K+1
311      WRITE(*,*) 'DRAWING VECTOR ',K
312      YE=4.5*DL+(1.0-TAUY)*0.5*DL+I*DY
313      XE=0
314      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
315      NDIR=0
316      VL=27.0/GS
317      CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
318      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
319      & ,S,MNAR,DS)
320      NDIR=0
321      VL=27.0/GS
322      XE=A
323      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
324      CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
325      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
326      & ,S,MNAR,DS)
327 15    CONTINUE
328    ELSE
329 C   ****
330 C   * DRAW SIDE 4 (X=-A/2) VECTORS. *
331 C   ****
332 C   ****
333 C   * END POINT CONTRIBUTIONS. *
334 C   ****
335      XMIN=A
336      XMAX=A+B
337      XE=(XMIN+XMAX)/2.0
338      NP=(NPY-1)/2
339      K=0
340      DO 16 I=0,NP-1
341      YE=4.5*DL+(1.0-TAUY)*0.5*DL+I*DY
342      K=K+1
343      WRITE(*,*) 'DRAWING VECTOR ',K
344      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
345      NDIR=0
346      VL=20.0/GS
347      CALL RVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
348      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
349      & ,S,MNAR,DS)
350      NDIR=1
351      VL=20.0/GS-S/GS
352      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)

```

```

353      & CALL RVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
354      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
355      & ,S,MNAR,DS)
356      VE=(I+1)*DY
357      K=K+1
358      WRITE(*,*) 'DRAWING VECTOR ',K
359      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
360      NDIR=0
361      VL=20.0/GS
362      CALL RVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
363      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
364      & ,S,MNAR,DS)
365      NDIR=1
366      VL=20.0/GS-S/GS
367      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
368      CALL RVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
369      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
370      & ,S,MNAR,DS)
371 16    CONTINUE
372 C ****
373 C * INTERIOR CONTRIBUTIONS.
374 C ****
375 DO 17 I=0,KM-2
376      YCE=(1.0-TAUY)*0.5*DL+0.5*DL+I*DL
377      DO 18 J=0,NPY-1
378      K=K+1
379      WRITE(*,*) 'DRAWING VECTOR ',K
380      YE=YCE+J*DY
381      NDIR=1
382      VL=20.0/GS-S/GS
383      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
384      CALL RVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
385      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
386      & ,S,MNAR,DS)
387      NDIR=0
388      VL=20.0/GS
389      CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T, ID)
390      CALL RVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
391      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
392      & ,S,MNAR,DS)
393 18    CONTINUE
394 17    CONTINUE
395 END IF
396 RETURN
397 END

1      SUBROUTINE RVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
2      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG
3      & ,S,MNAR,DS)
4 C ****
5 C * THIS SUBROUTINE DRAWS A VECTOR LINE.
6 C ****
7 C * INPUTS:
8 C *
9 C *   XE,YE - FIRST COORDINATE OF VECTOR LINE.
10 C *   MN - MAXIMUM NUMBER OF COORDINATE POINTS.
11 C *   MT - MAXIMUM NUMBER OF TRIANGLES.
12 C *   N2T(MT,3) - NODE MATRIX WHERE FIRST INDEX REPRESENTS
13 C *                 EACH TRIANGLE AND SECOND INDEX REPRESENTS

```

```

14 C      * THE NODE NUMBERS.
15 C      * T2N(MN,MP) - MATRIX OF TRIANGLE TO NODE CONNECTIONS
16 C      * WHERE THE FIRST INDEX IS THE NODE NUMBER
17 C      * AND THE SECOND IS THE LOCAL TRIANGLE NUMBER.
18 C      * X(MN) - COORDINATE VECTORS OF GRID.
19 C      * Y(MN)
20 C      * FX(MN) - VECTOR FUNCTION FIELD VALUES AT EACH COORDINATE.
21 C      * FY(MN)
22 C      * AX,BX - GRAPHICAL SCALE CONSTANTS.
23 C      * AY,BY
24 C      * VL - SPACING BETWEEN ARROWHEADS IN REAL COORDINATES.
25 C      * BANG - ARROW APEX HALF ANGLE IN DEGREES.
26 C      * S - ARROW HEAD SIDE LENGTH IN POINT SIZE.
27 C      * MNAR - MAXIMUM NUMBER OF ARROWS ALLOWED.
28 C      * DS - STEP SIZE.
29 C      *
30 C      * OUTPUTS:
31 C      *
32 C      **** ****
33 REAL*4 X(MN),Y(MN),FX(MN),FY(MN)
34 INTEGER N2T(MT,3),T2N(MN,MP)
35 C      **** ****
36 C      * STORE THE FIRST TRIANGLE NUMBER.
37 C      **** ****
38 IFIRST=ID
39 C      **** ****
40 C      * STORE THE FIRST POINT.
41 C      **** ****
42 AWIDTH=1.0
43 BWIDTH=AWIDTH/2.0
44 XFIRST=XE
45 YFIRST=YE
46 C      **** ****
47 C      * TRANSFORM THE INITIAL POINT.
48 C      **** ****
49 CALL MPRPNT(XE,YE,AWIDTH,BWIDTH,XAT,YAT,ZAT)
50 CALL TRFPNT(XAT,YAT,ZAT,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
51 WRITE(1,100) XAT,YAT,' moveto'
52 C      **** ****
53 C      * BEGIN THE VECTOR SEGMENTS.
54 C      **** ****
55 K=0
56 VSUM=0.0
57 KAR=0
58 999 CONTINUE
59 K=K+1
60 C      WRITE(*,*) 'SEGMENT NUMBER= ',K,' TRIANGLE NUMBER= ',ID
61 C      **** ****
62 C      * IF THIS IS NOT THE FIRST LINE THEN DETERMINE WHAT TRIANGLE
63 C      * THE DRAWN LINE CONNECTS TO BY LOOKING AT THE TWO NODES NA AND
64 C      * NB FORMING THE LAST INTERSECTION.
65 C      **** ****
66 IF (K .GT. 1) THEN
67   IF (K .GT. 300) THEN
68     GO TO 99
69   ELSE
70   END IF
71   IF (ID .EQ. 0) THEN
72 C      **** ****

```

```

73 C * THERE IS NO NEXT TRIANGLE WHICH MEANS THE VECTOR *
74 C * TERMINATES ON A BOUNDARY. *
75 C ****
76 WRITE(*,*) 'TERMINATE ON BOUNDARY TRIANGLE'
77 GO TO 99
78 ELSE IF (ID .EQ. IFIRST) THEN
79 C WRITE(*,*) 'BACK TO ORIGINAL TRIANGLE'
80 C GO TO 99
81 ELSE
82 END IF
83 XA=XF
84 YA=YF
85 ELSE
86 XA=XE
87 YA=YE
88 END IF
89 C ****
90 C * FIND THE VECTOR FUNCTION VALUE AT THE POINT (XA,YA). *
91 C ****
92 CALL VECFCOF(MN,MT,ID,X,Y,XA,YA,N2T,FX,FY,A,B,E,C,D,F)
93 C ****
94 C * GET THE VECTOR COMPONENTS AT THE POINT (XA,YA). *
95 C ****
96 FYA=A*XA+B*YA+E
97 FXA=C*XA+D*YA+F
98 FM=SQRT(FXA*FXA+FYA*FYA)
99 FXN=FXA/FM
100 FYN=FYA/FM
101 IF (NDIR .EQ. 0) THEN
102 XF=XA+DS*FXN
103 YF=YA+DS*FYN
104 ELSE
105 XF=XA-DS*FXN
106 YF=YA-DS*FYN
107 END IF
108 C ****
109 C * GET THE NEXT TRIANGLE NUMBER. *
110 C ****
111 IDOLD=ID
112 CALL FDSTRI(XF,YF,MN,X,Y,MT,MP,NT,N2T,T2N,IDL0D, ID)
113 C WRITE(*,*) 'ITERATION= ',K,IDL0D, ID,XF,YF,FXA,FYA
114 IF (ID .EQ. IFIRST) THEN
115 C XF=XFIRST
116 C YF=YFIRST
117 ELSE IF (ID .EQ. 0) THEN
118 C WRITE(*,*) 'BOUNDARY NODE'
119 C ****
120 C * ADJUST THE POINT IF IT IS OUTSIDE THE BOUNDARY. *
121 C ****
122 IF (XF .LT. XMIN) THEN
123 XF=XMIN
124 ELSE IF (XF .GT. XMAX) THEN
125 XF=XMAX
126 ELSE
127 END IF
128 IF (YF .LT. YMIN) THEN
129 YF=YMIN
130 ELSE IF (YF .GT. YMAX) THEN
131 YF=YMAX

```

```

132      ELSE
133      END IF
134 C      *****
135 C      * DRAW THE LINE FROM (XA,YA) TO (XF,YF). *
136 C      *****
137      CALL MPRPNT(XF,YF,AWIDTH,BWIDTH,XFT,YFT,ZFT)
138      CALL TRFPNT(XFT,YFT,ZFT,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
139      WRITE(1,100) XFT,YFT,' lineto'
140      GO TO 99
141      END IF
142 C      *****
143 C      * ADJUST THE POINT IF IT IS OUTSIDE THE BOUNDARY. *
144 C      *****
145      IF (XF .LT. XMIN) THEN
146          XF=XMIN
147      ELSE IF (XF .GT. XMAX) THEN
148          XF=XMAX
149      ELSE
150      END IF
151      IF (YF .LT. YMIN) THEN
152          YF=YMIN
153      ELSE IF (YF .GT. YMAX) THEN
154          YF=YMAX
155      ELSE
156      END IF
157 C      *****
158 C      * DRAW THE LINE FROM (XA,YA) TO (XF,YF). *
159 C      *****
160 C      *****
161 C      * TRANSFORM THE POINT. *
162 C      *****
163      CALL MPRPNT(XF,YF,AWIDTH,BWIDTH,XFT,YFT,ZFT)
164      CALL TRFPNT(XFT,YFT,ZFT,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
165      WRITE(1,100) XFT,YFT,' lineto'
166 C      *****
167 C      * DRAW ARROW OR ARROWS IF APPROPRIATE. *
168 C      *****
169      DAR=SQRT((XA-XF)*(XA-XF)+(YA-YF)*(YA-YF))
170      IF (DAR .GT. 0.0) THEN
171          IF (NDIR .EQ. 0) THEN
172              CALL RARR3D(MNAR,VL,BANG,S,XA,YA,XF,YF,FXN,FYN,VSUM,KAR
173                                     ,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
174          ELSE
175              CALL RARR3D(MNAR,VL,BANG,S,XF,YF,XA,YA,FXN,FYN,VSUM,KAR
176                                     ,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
177          END IF
178      ELSE
179      END IF
180      IF (FM .LT. .001) THEN
181          WRITE(*,*) 'CLOSE TO AN UNKNOWN CRITICAL POINT'
182          GO TO 99
183      ELSE
184      END IF
185      GO TO 999
186 C      *****
187 C      * STOP. *
188 C      *****
189 99      CONTINUE
190      WRITE(1,*) 'stroke'

```

```

191 100  FORMAT(2(F6.2,1X),A7)
192  ID=IFIRST
193  RETURN
194  END

1      SUBROUTINE RARR3D(MNAR,VL,BANG,S,XA,YA,XF,YF,FXN,FYN,VSUM,KAR
2      &                               ,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
3 C  ****
4 C  * THIS SUBROUTINE CONTROLS THE DRAWING OF ARROWHEADS ON A LINE. *
5 C  ****
6 C  * INPUTS: *
7 C  *
8 C  *   MNAR    - MAXIMUM NUMBER OF ARROWS ALLOWED. *
9 C  *   VL      - SPACING BETWEEN ARROWHEADS IN REAL COORDINATES. *
10 C  *   BANG    - ARROW APEX HALF ANGLE IN DEGREES. *
11 C  *   S       - ARROW HEAD SIDE LENGTH IN POINT SIZE. *
12 C  *   XA, YA  - BASE OF ARROW. *
13 C  *   XF, YF  - TIP OF ARROW. *
14 C  *   FXN, FYN - NORMALIZED VECTOR FUNCTION VALUES. *
15 C  *   VSUM    - ACCUMULATED LINE LENGTH. *
16 C  *   KAR     - NUMBER OF ARROWS DRAWN SO FAR. *
17 C  *   AX,BX   - GRAPHICAL SCALE CONSTANTS. *
18 C  *   AY,BY   - *
19 C  ****
20  D=SQRT((XA-XF)*(XA-XF)+(YA-YF)*(YA-YF))
21  STEP=VSUM+D
22  IF (STEP .GT. VL) THEN
23    KAR=KAR+1
24    IF (KAR .GT. MNAR) THEN
25      GO TO 2
26    ELSE
27    END IF
28 C  ****
29 C  * THE NEXT LINE SEGMENT IS LONGER THAN THE ARROW SPACING. *
30 C  * SO AN ARROW SHOULD BE DRAWN. *
31 C  ****
32  RAT=(VL-VSUM)/D
33  XFW=XA+RAT*(XF-XA)
34  YFW=YA+RAT*(YF-YA)
35 C  ****
36 C  * DRAW THE ARROW. *
37 C  ****
38  CALL RAHD3D(BANG,GS,S,XFW,YFW,FXN,FYN,XD,YD,ZD,C1,C2
39  &                               ,C3,C4,C5,X0,Y0)
40 C  ****
41 C  * COMPUTE THE AMOUNT LEFT OVER AFTER THE ARROW IS DRAWN. *
42 C  ****
43  DLT=SQRT((XF-XFW)*(XF-XFW)+(YF-YFW)*(YF-YFW))
44  IF (DLT .LT. VL) THEN
45 C  ****
46 C  * NO MORE ARROWS ARE DRAWN ON THIS SEGMENT. *
47 C  ****
48  VSUM=DLT
49  ELSE
50 C  ****
51 C  * MORE THAN ONE ARROW IS DRAWN ON THIS SEGMENT. *
52 C  ****
53  NAR=DLT/VL
54  DO 1 I=1,NAR

```

```

55          KAR=KAR+1
56          IF (KAR .GT. MNAR) THEN
57              GO TO 2
58          ELSE
59          END IF
60          RAT=(VL-VSUM+I*VL)/D
61          XFW=XA+RAT*(XF-XA)
62          YFW=YA+RAT*(YF-YA)
63          CALL RAHD3D(BANG,GS,S,XFW,YFW,FXN,FYN,XD,YD,ZD,C1,C2
64          & ,C3,C4,C5,X0,Y0)
65      1    CONTINUE
66          VSUM=SQRT((XF-XFW)*(XF-XFW)+(YF-YFW)*(YF-YFW))
67      END IF
68      ELSE
69 C      ****
70 C      * THE LINE LENGTH PLUS THE ACCUMULATED DISTANCE IS STILL LESS *
71 C      * THAN THE ARROW SPACING. NO ARROW SHOULD BE DRAWN JUST        *
72 C      * ACCUMULATE THE DISTANCE.                                     *
73 C      ****
74          VSUM=VSUM+D
75      END IF
76      2    CONTINUE
77      RETURN
78  END

1      SUBROUTINE RAHD3D(BANG,GS,S,X2,Y2,FXN,FYN,XD,YD,ZD,C1,C2
2      &,C3,C4,C5,X0,Y0)
3 C      ****
4 C      * THIS SUBROUTINE DRAWS A ARROW HEAD FROM (X1,Y1) TO (X2,Y2).   *
5 C      ****
6 C      * INPUTS:                                         *
7 C      *                                         *
8 C      *     BANG - ARROW APEX HALF ANGLE IN DEGREES.           *
9 C      *     S      - ARROWHEAD SIDE LENGTH.                   *
10 C      *    X2,Y2 - TIP OF ARROW.                         *
11 C      *    FXN,FYN - NORMALIZED VECTOR FUNCTION VALUES.   *
12 C      ****
13 RAD=.17453293E-01
14 X1=X2-FXN
15 Y1=Y2-FYN
16 D=SQRT((X2-X1)*(X2-X1)+(Y2-Y1)*(Y2-Y1))
17 A=1.0-S/(GS*D)
18 X3=X1+A*(X2-X1)
19 Y3=Y1+A*(Y2-Y1)
20 C      ****
21 C      * TRANSLATE THE ORIGIN TO (X2,Y2) AND ROTATE (X3,Y3) PLUS AND   *
22 C      * MINUS BANG DEGREES TO GENERATE THE VERTICES OF THE ARROW.   *
23 C      ****
24 XA=X3-X2
25 YA=Y3-Y2
26 ARG=RAD*BANG
27 CA=COS(ARG)
28 SA=SIN(ARG)
29 XCA=XA*CA
30 XSA=XA*SA
31 YCA=YA*CA
32 YSA=YA*SA
33 X4=X2+XCA-YSA
34 Y4=Y2+XSA+YCA

```

```

35      X5=X2+XCA+YSA
36      Y5=Y2-XSA+YCA
37 c      X6=(X4+X5)/2.0
38 c      Y6=(Y4+Y5)/2.0
39 C      WRITE(1,*) X1,Y1,' moveto'
40 C      WRITE(1,*) X6,Y6,' lineto stroke'
41 C      ****
42 C      * TRANSLATE THE ORIGIN TO (X2,Y2) AND ROTATE (X3,Y3) PLUS AND *
43 C      * MINUS BANG DEGREES TO GENERATE THE VERTICES OF THE ARROW *
44 C      ****
45      AWIDTH=1.0
46      BWIDTH=0.5
47      CALL MPRPNT(X2,Y2,AWIDTH,BWIDTH,X2T,Y2T,Z2T)
48      CALL MPRPNT(X4,Y4,AWIDTH,BWIDTH,X4T,Y4T,Z4T)
49      CALL MPRPNT(X5,Y5,AWIDTH,BWIDTH,X5T,Y5T,Z5T)
50      CALL TRFPNT(X2T,Y2T,Z2T,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
51      CALL TRFPNT(X4T,Y4T,Z4T,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
52      CALL TRFPNT(X5T,Y5T,Z5T,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
53      WRITE(1,*) 'gsave newpath'
54      WRITE(1,*) X2T,Y2T,' moveto ',X4T,Y4T,' lineto '
55      &           ,X5T,Y5T,' lineto'
56      WRITE(1,*) 'closepath 0 setgray fill stroke grestore'
57      RETURN
58      END

1      SUBROUTINE MPRPNT(U,V,A,B,X,Y,Z)
2 C      ****
3 C      * THIS SUBROUTINE MAPS A POINT FROM THE UNFOLDED WAVEGUIDE *
4 C      * (U,V) SPACE TO THE (X,Y,Z) FOLDED WAVEGUIDE SPACE.          *
5 C      ****
6 C      * INPUTS:                                     *
7 C      *                                     *
8 C      *     U,V - COORDINATES OF THE UNFOLDED WAVEGUIDE POINT.   *
9 C      *     A    - X DIMENSION OF WAVEGUIDE SUCH THAT -A/2 <= X <= A/2. *
10 C      *    B    - X DIMENSION OF WAVEGUIDE SUCH THAT -B/2 <= Y <= B/2. *
11 C      *                                     *
12 C      * OUTPUTS:                                     *
13 C      *                                     *
14 C      *     X,Y,Z - COORDINATES OF MAPPED POINT.                 *
15 C      ****
16      IF ((U .GE. -A-B) .AND. (U .LT. -B)) THEN
17          X=U+B+A/2.0
18          Y=B/2.0
19      ELSE IF ((U .GE. -B) .AND. (U .LT. 0.0)) THEN
20          X=A/2.0
21          Y=-U-B/2.0
22      ELSE IF ((U .GE. 0.0) .AND. (U .LT. A)) THEN
23          X=-U+A/2.0
24          Y=-B/2.0
25      ELSE
26          X=-A/2.0
27          Y=U-A-B/2.0
28      END IF
29      Z=V
30      RETURN
31      END

```

```

1      PROGRAM FLDCIR
2 C ****
3 C * THIS PROGRAM GENERATES A VECTOR PLOT OF THE TRIANGULARIZED   *
4 C * REGION OF A CIRCULAR WAVEGUIDE.                                *
5 C ****
6 C * TIMOTHY J. PETERS          LAST UPDATED   *
7 C * THE AEROSPACE CORPORATION    5/12/92        *
8 C * 2350 EAST EL SEGUNDO BOULEVARD   *
9 C * EL SEGUNDO, CA 90245          *
10 C ****
11 C PARAMETER (MN=6000,MT=12000,MP=6,NC=5,NL=6)
12 C REAL*4 U(MN),V(MN),FU(MN),FV(MN),PHI(50),CA(NC),CB(NC)
13 C REAL*4 XQ(3),YQ(3),ZQ(3)
14 C REAL*8 ARG
15 C INTEGER LC(NC)
16 C CHARACTER*30 LABEL(NL)
17 C INTEGER*4 N2T(MT,3),T2N(MN,MP)
18 C PI=.3141593E+01
19 C TP=.6283153E+01
20 C RAD=.17453293E-01
21 C ****
22 C * INPUTS:               *
23 C *               *
24 C * GS - GRAPH SCALE CONSTANT.          *
25 C * PO - PHI OBSERVATION ANGLE (DEGREES).   *
26 C * TO - THETA OBSERVATION ANGLE (DEGREES).  *
27 C ****
28 C PO=50.0
29 C TO=60.0
30 C ****
31 C * READ THE COORDINATE DATA AND VECTOR FUNCTION DATA FROM A FILE. *
32 C ****
33 C     OPEN(UNIT=1,FILE='CUVFDF')
34 C     READ(1,*) A,C,KM,N,M
35 C     READ(1,*) NU,NV
36 C     READ(1,*) NN
37 C     VMAX=0.0
38 C     DO 1 I=1,NN
39 C       READ(1,*) U(I),V(I),FU(I),FV(I)
40 C     VMAG=SQRT(FU(I)*FU(I)+FV(I)*FV(I))
41 C     IF (VMAG .GT. VMAX) THEN
42 C       VMAX=VMAG
43 C     ELSE
44 C     END IF
45 C     1 CONTINUE
46 C     CLOSE(1)
47 C     WRITE(*,*) NN,' DATA POINTS READ IN'
48 C ****
49 C * NORMALIZE THE VECTOR COMPONENTS SO THAT THE MAXIMUM MAGNITUDE   *
50 C * IS 1.0.                                *
51 C ****
52 C     DO 2 I=1,NN
53 C       FU(I)=FU(I)/VMAX
54 C       FV(I)=FV(I)/VMAX
55 C     2 CONTINUE
56 C ****
57 C * READ THE TRIANGLE NODE DATA.          *
58 C ****
59 C     OPEN(UNIT=1,FILE='N2TDAT')

```

```

60      READ(1,*) NT
61      DO 3 I=1,NT
62          READ(1,*) N2T(I,1),N2T(I,2),N2T(I,3)
63 3    CONTINUE
64      CLOSE(1)
65      WRITE(*,*) 'NODE TO TRIANGLE DATA READ IN'
66 C ****
67 C * READ THE TRIANGLE TO NODE CONNECTION DATA. *
68 C ****
69 OPEN(UNIT=1,FILE='T2NDAT')
70 READ(1,*) NN
71 DO 4 I=1,NN
72     READ(1,*) (T2N(I,J),J=1,MP)
73 4    CONTINUE
74      CLOSE(1)
75      WRITE(*,*) 'TRIANGLE TO NODE DATA READ IN'
76 C ****
77 C * COMPUTE CENTROID OF GRAPH. *
78 C ****
79 GS=180.0
80 XMIN=-A/2.0
81 XMAX=A/2.0
82 YMIN=-A/2.0
83 YMAX=A/2.0
84 ZMIN=0.0
85 ZMAX=C
86 XC=(XMIN+XMAX)/2.0
87 YC=(YMIN+YMAX)/2.0
88 ZC=(ZMIN+ZMAX)/2.0
89 C ****
90 C * CONVERT THE OBSERVATION ANGLES TO RADIANS. *
91 C ****
92 RAD=.17453293E-01
93 PI=.3141593E+01
94 PI4=PI/4.0
95 XD=XC
96 YD=YC
97 ZD=ZC
98 CALL SCALE(GS,XD,YD,ZD)
99 C ****
100 C * CONVERT THE OBSERVATION ANGLES TO RADIANS. *
101 C ****
102 POBS=RAD*PO
103 TOBS=RAD*TO
104 C ****
105 C * COMPUTE THE ROTATION ANGLES WHICH YIELD THE DESIRED OBSERVATION*
106 C * ANGLES. *
107 C ****
108 RP=-POBS-PI/2.0
109 RT=TOBS
110 C ****
111 C * COMPUTE THE CONSTANTS FOR THE ROTATION MATRIX. *
112 C ****
113 CALL RCOEFF(RP,RT,C1,C2,C3,C4,C5)
114 C ****
115 C * COMPUTE THE CENTER OF THE PRINTING DEVICE PAGE. *
116 C ****
117 X0=72*8.5/2.0
118 Y0=72*11.0/2.0

```

```

119 C ****
120 C * SET THE GRAPHIC BOUNDING BOX. *
121 C ****
122 WX=(XMAX-XMIN)
123 WY=(YMAX-YMIN)
124 WIDTH=GS*WX
125 HEIGHT=GS*WY
126 UA=XO-WIDTH/2.0-48.0
127 UB=XO+WIDTH/2.0+91.0
128 VA=YO-HEIGHT/2.0-243.0
129 VB=YO+HEIGHT/2.0+239.0
130 C ****
131 C * COMPUTE THE BOUNDING BOX FOR THE POSTSCRIPT IN THE DEFAULT *
132 C * COORDINATE SYSTEM. *
133 C ****
134 OPEN(UNIT=1,FILE='FoldedCirc.ps')
135 REWIND 1
136 WRITE(1,*) '%!PS-Adobe-1.0'
137 WRITE(1,*) '%%%Creator: Timothy J. Peters'
138 WRITE(1,*) '%%%Title: Graph'
139 WRITE(1,*) '%%%CreationDate: 5-14-92'
140 WRITE(1,100) '%%%BoundingBox:',INT(UA),INT(VA),INT(UB),INT(VB)
141 100 FORMAT(A14,4(1X,I3))
142 WRITE(1,*) '%%%EndComments'
143 WRITE(1,*) '/dot2 {2 0 360 arc 0 setgray fill stroke} def'
144 WRITE(1,*) '/dot3 {3 0 360 arc 0 setgray fill stroke} def'
145 WRITE(1,*) '/dot4 {4 0 360 arc 0 setgray fill stroke} def'
146 WRITE(1,*) '/black {0 0 0 setrgbcolor} def'
147 WRITE(1,*) '/white {1 1 1 setrgbcolor} def'
148 WRITE(1,*) '/gray-lt {0.92 0.92 0.92 setrgbcolor} def'
149 WRITE(1,*) '/gray-lt-med {0.65 0.65 0.65 setrgbcolor} def'
150 WRITE(1,*) '/gray {0.45 0.45 0.45 setrgbcolor} def'
151 WRITE(1,*) '/gray-dk-med {0.3 0.3 0.3 setrgbcolor} def'
152 WRITE(1,*) '/gray-dk {0.14 0.14 0.14 setrgbcolor} def'
153 WRITE(1,*) '/red {1 0 0 setrgbcolor} def'
154 WRITE(1,*) '/magenta {1 0 1 setrgbcolor} def'
155 WRITE(1,*) '/green {0 1 0 setrgbcolor} def'
156 WRITE(1,*) '/blue {0 0 1 setrgbcolor} def'
157 WRITE(1,*) '/cyan {0 1 1 setrgbcolor} def'
158 WRITE(1,*) '/yellow {1 1 0 setrgbcolor} def'
159 WRITE(1,*) '/orange {1 0.5 0 setrgbcolor} def'
160 WRITE(1,*) '/brown {0.5 0.5 0 setrgbcolor} def'
161 WRITE(1,*) '/kakhi {0.5 1 0 setrgbcolor} def'
162 WRITE(1,*) '/blue-lt {0.5 1 1 setrgbcolor} def'
163 WRITE(1,*) '/green-lt {0.5 1 0.5 setrgbcolor} def'
164 WRITE(1,*) '/green-blue {0 1 0.5 setrgbcolor} def'
165 WRITE(1,*) '/purple {0.6 0 1.0 setrgbcolor} def'
166 WRITE(1,*) '/filtri {moveto lineto lineto'
167 WRITE(1,*) ' closepath fill stroke} def'
168 WRITE(1,*) '/filqud {moveto lineto lineto lineto closepath fill'
169 WRITE(1,*) ' stroke} def'
170 WRITE(1,*) '/filpnt {moveto lineto lineto lineto'
171 WRITE(1,*) ' lineto closepath fill stroke} def'
172 WRITE(1,*) '%%%EndProlog'
173 WRITE(1,*) '1 setlinejoin .4 setlinewidth'
174 C ****
175 C * DRAW THE EPS BOUNDING BOX. *
176 C ****
177 C WRITE(1,*) 'gsave 0.2 setlinewidth'

```

```

178 C      WRITE(1,*) UA,VA,' moveto'
179 C      WRITE(1,*) UB,VA,' lineto'
180 C      WRITE(1,*) UB,VB,' lineto'
181 C      WRITE(1,*) UA,VB,' lineto'
182 C      WRITE(1,*) 'closepath stroke grestore'
183 C      ****
184 C      * ASSIGN CONTOUR VALUES AND LABELS. *
185 C      ****
186 CA(1)=0.0
187 CB(1)=0.15
188 CA(2)=0.15
189 CB(2)=0.4
190 CA(3)=0.4
191 CB(3)=0.6
192 CA(4)=0.6
193 CB(4)=0.8
194 CA(5)=0.8
195 CB(5)=1.0
196 LABEL(1)='0'
197 LABEL(2)='15'
198 LABEL(3)='4'
199 LABEL(4)='6'
200 LABEL(5)='8'
201 LABEL(6)='1'
202 C      ****
203 C      * ASSIGN THE COLORS. SEE SUBROUTINE SETCOL FOR COLOR CHOICES. *
204 C      ****
205 C GRAY SCALE
206 LC(1)=2
207 LC(2)=3
208 LC(3)=4
209 LC(4)=5
210 LC(5)=6
211 C COLOR
212 c      LC(1)=18
213 c      LC(2)=14
214 c      LC(3)=12
215 c      LC(4)=16
216 c      LC(5)=13
217 IF (P0 .LE. 90.0) THEN
218     XCMIN=(RAD*(P0+90.0))*A
219     XCMAX=(RAD*(P0+270.0))*A
220 ELSE
221 END IF
222 XMIN=0.0
223 XMAX=TP*A
224 YMIN=0.0
225 YMAX=C
226 C      ****
227 C      * DRAW THE BACK SIDE OF THE CYLINDER. *
228 C      ****
229 KUS=INT(1+(NU-1)*(P0+90.0)/360.0)
230 ITMIN=2*(KUS-1)*(NV-1)+1
231 KUSM=KUS+(NU+1)/2
232 ITMAX=2*(KUSM-1)*(NV-1)
233 C      ****
234 C      * PROCESS TRIANGLES. *
235 C      ****
236 DO 5 I=ITMIN,ITMAX

```

```

237      N1=N2T(I,1)
238      N2=N2T(I,2)
239      N3=N2T(I,3)
240      F1=SQRT(FU(N1)*FU(N1)+FV(N1)*FV(N1))
241      F2=SQRT(FU(N2)*FU(N2)+FV(N2)*FV(N2))
242      F3=SQRT(FU(N3)*FU(N3)+FV(N3)*FV(N3))
243      CALL MPCPNT(U(N1),V(N1),A,XQ(1),YQ(1),ZQ(1))
244      CALL MPCPNT(U(N2),V(N2),A,XQ(2),YQ(2),ZQ(2))
245      CALL MPCPNT(U(N3),V(N3),A,XQ(3),YQ(3),ZQ(3))
246 C      ****
247 C      * TRANSFORM THE 3 VERTICES OF THE TRIANGLE. *
248 C      ****
249      CALL TRFTRI(XQ,YQ,ZQ,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
250 C      ****
251 C      * DRAW THE CONTOUR. *
252 C      ****
253      CALL CONTUR(XQ(1),YQ(1),XQ(2),YQ(2),XQ(3),YQ(3),F1,F2,F3
254      ,CA,CB,NC,LC)
255 5    &   CONTINUE
256 C      ****
257 C      * DRAW THE VECTORS ON THE BACK SIDE. *
258 C      ****
259      ICLIP=1
260      BANG=20.0
261      S=8.0
262      MNAR=1
263      DS=.003
264      XMIN=0.0
265      XMAX=TP*A
266      YMIN=0.0
267      YMAX=C
268      WRITE(1,*) '0 setgray'
269      CALL CWTE3D(MN,MT,MP,NT,N2T,T2N,U,V,A,C,KM,XMIN,XMAX,YMIN,YMAX
270      &,FU,FV,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,BANG,S,MNAR,DS
271      &,XCMIN,XCMAX,ICLIP)
272 C      ****
273 C      * DRAW THE FRONT SIDE OF THE CYLINDER. *
274 C      ****
275      DO 6 I=ITMAX+1,NT
276          N1=N2T(I,1)
277          N2=N2T(I,2)
278          N3=N2T(I,3)
279          F1=SQRT(FU(N1)*FU(N1)+FV(N1)*FV(N1))
280          F2=SQRT(FU(N2)*FU(N2)+FV(N2)*FV(N2))
281          F3=SQRT(FU(N3)*FU(N3)+FV(N3)*FV(N3))
282          CALL MPCPNT(U(N1),V(N1),A,XQ(1),YQ(1),ZQ(1))
283          CALL MPCPNT(U(N2),V(N2),A,XQ(2),YQ(2),ZQ(2))
284          CALL MPCPNT(U(N3),V(N3),A,XQ(3),YQ(3),ZQ(3))
285 C      ****
286 C      * TRANSFORM THE 3 VERTICES OF THE TRIANGLE. *
287 C      ****
288      CALL TRFTRI(XQ,YQ,ZQ,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
289 C      ****
290 C      * DRAW THE CONTOUR. *
291 C      ****
292      CALL CONTUR(XQ(1),YQ(1),XQ(2),YQ(2),XQ(3),YQ(3),F1,F2,F3
293      ,CA,CB,NC,LC)
294 6    &   CONTINUE
295      DO 7 I=1,ITMIN-1

```

```

296      N1=N2T(I,1)
297      N2=N2T(I,2)
298      N3=N2T(I,3)
299      F1=SQRT(FU(N1)*FU(N1)+FV(N1)*FV(N1))
300      F2=SQRT(FU(N2)*FU(N2)+FV(N2)*FV(N2))
301      F3=SQRT(FU(N3)*FU(N3)+FV(N3)*FV(N3))
302      CALL MPCPNT(U(N1),V(N1),A,XQ(1),YQ(1),ZQ(1))
303      CALL MPCPNT(U(N2),V(N2),A,XQ(2),YQ(2),ZQ(2))
304      CALL MPCPNT(U(N3),V(N3),A,XQ(3),YQ(3),ZQ(3))
305 C      ****
306 C      * TRANSFORM THE 3 VERTICES OF THE TRIANGLE. *
307 C      ****
308      CALL TRFTRI(XQ,YQ,ZQ,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
309 C      ****
310 C      * DRAW THE CONTOUR. *
311 C      ****
312      CALL CONTUR(XQ(1),YQ(1),XQ(2),YQ(2),XQ(3),YQ(3),F1,F2,F3
313      ,CA,CB,NC,LC)
314      & CONTINUE
315 C      ****
316 C      * DRAW THE VECTORS ON THE FRONT SIDE. *
317 C      ****
318      ICLIP=2
319      XMIN=0.0
320      XMAX=TP*A
321      YMIN=0.0
322      YMAX=C
323      BANG=20.0
324      S=8.0
325      MNAR=1
326      DS=.003
327      WRITE(1,*) '0 setgray'
328      CALL CWTE3D(MN,MT,MP,NT,N2T,T2N,U,V,A,C,KM,XMIN,XMAX,YMIN,YMAX
329      & ,FU,FV,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,BANG,S,MNAR,DS
330      & ,XCMIN,XCMAX,ICLIP)
331 C      ****
332 C      * DRAW THE LEGEND. *
333 C      ****
334      SL=15.0
335      SH=30.0
336      DS=0.0
337      ITYPE=1
338      US=409.0
339      VS=155.0
340      CALL LEGEND(US,VS,NC,NL,LC,LABEL,SL,SH,DS,ITYPE)
341 C      ****
342 C      * DRAW THE PAGE. *
343 C      ****
344      WRITE(1,*) 'showpage'
345      CLOSE(1)
346      END

1      SUBROUTINE CWTE3D(MN,MT,MP,NT,N2T,T2N,X,Y,A,C,KM,XMIN,XMAX,YMIN
2      & ,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,BANG,S,MNAR,DS
3      & ,XCMIN,XCMAX,ICLIP)
4 C      ****
5 C      * THIS SUBROUTINE DRAWS VECTOR LINES FOR A CIRCULAR *
6 C      * WAVEGUIDE WITH TE11 MODE FIELDS. *
7 C      ****

```

```

8 C * INPUTS: *
9 C *
10 C * MN - MAXIMUM NUMBER OF COORDINATE POINTS. *
11 C * MT - MAXIMUM NUMBER OF TRIANGLES. *
12 C * MP - MAXIMUM NUMBER OF TRIANGLES CONNECTED TO A SINGLE *
13 C * NODE. *
14 C * NT - NUMBER OF TRIANGLES. *
15 C * N2T(MT,3) - NODE MATRIX WHERE FIRST INDEX REPRESENTS *
16 C * EACH TRIANGLE AND SECOND INDEX REPRESENTS *
17 C * THE NODE NUMBERS. *
18 C * T2N(MN,MP) - MATRIX OF TRIANGLE TO NODE CONNECTIONS *
19 C * WHERE THE FIRST INDEX IS THE NODE NUMBER *
20 C * AND THE SECOND IS THE LOCAL TRIANGLE NUMBER. *
21 C * X(MN) - COORDINATE VECTORS OF GRID. *
22 C * Y(MN)
23 C * A - RADIUS OF CIRCULAR WAVEGUIDE (WAVELENGTHS). *
24 C * C - LENGTH OF CIRCULAR WAVEGUIDE (WAVELENGTHS). *
25 C * XMIN,XMAX - DIMENSIONS OF UNFOLDED CIRCULAR WAVEGUIDE *
26 C * YMIN,YMAX (WAVELENGTHS). *
27 C * FX(MN) - VECTOR FUNCTION FIELD VALUES AT EACH COORDINATE. *
28 C * FY(MN)
29 C * GS - GRAPHICAL SCALE CONSTANT. *
30 C * BX,BY - GRAPHICAL OFFSETS. *
31 C * VL - SPACING BETWEEN ARROWHEADS IN REAL COORDINATES. *
32 C * BANG - ARROW APEX HALF ANGLE IN DEGREES. *
33 C * S - ARROW HEAD SIDE LENGTH IN POINT SIZE. *
34 C * MNAR - MAXIMUM NUMBER OF ARROWS ALLOWED. *
35 C * DS - STEP SIZE. *
36 C * XCMIN,XCMAX - HORIZONTAL CLIPPING BOUNDARY (WAVELENGTHS) *
37 C * ICLIP - ICLIP=1 MEANS THAT ALL LINES OUTSIDE THE CLIPPING *
38 C * BOUNDARY ARE CLIPPED. ICLIP=2 MEANS THAT ALL *
39 C * LINES INSIDE THE CLIPPING BOUNDARY ARE CLIPPED. *
40 C ****
41 REAL*4 X(MN),Y(MN),FX(MN),FY(MN)
42 INTEGER N2T(MT,3),T2N(MN,MP)
43 PI=.3141593E+01
44 TP=.6283185E+01
45 RAD=.17453293E-01
46 K=0
47 C ****
48 C * SET SPACING PARAMETERS BASED ON CRITICAL POINT LOCATIONS. *
49 C ****
50 XWIDTH=PI*A
51 TAUX=0.84
52 NPX=9
53 DX=TAUX*XWIDTH/(NPX-1)
54 TAUY=0.9
55 NPY=5
56 DY=(TAUY*C/KM)/(NPY-1)
57 C ****
58 C * SET GLOBAL VECTOR INPUTS. *
59 C ****
60 AH=S*COS(RAD*BANG)
61 C ****
62 C * LEFT REGION HORIZONTAL LINES. *
63 C ****
64 XE=0.0
65 DO 1 I=0,KM
66 YZ=-C/(2*KM)+I*C/KM

```

```

67      DO 2 J=0,NPY-1
68      YE=YZ+(C/KM)*(1.0-TAUY)/2.0+J*DY
69      IF ((YE .GT. DY/2.0) .AND. (YE .LT. C-DY/2.0)) THEN
70          K=K+1
71          WRITE(*,*) 'DRAWING VECTOR ',K
72          CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
73          NDIR=0
74          VL=20.0/GS
75          WRITE(1,*) '% begin direction 1 of vector ',K
76          CALL CVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
77          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG,S
78          & ,MNAR,DS,XCMIN,XCMAX,ICLIP)
79          WRITE(1,*) '% end direction 1 of vector ',K
80          NDIR=1
81          VL=20.0/GS-S/GS
82          WRITE(1,*) '% begin direction 2 of vector ',K
83 C          CALL CVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
84 C          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,VL,BANG,S,MNAR,DS
85 C          & ,XCMIN,XCMAX,ICLIP)
86          CALL CVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
87          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG,S
88          & ,MNAR,DS,XCMIN,XCMAX,ICLIP)
89          WRITE(1,*) '% end direction 2 of vector ',K
90          ELSE
91          END IF
92 2      CONTINUE
93 1      CONTINUE
94 C      *****
95 C      * LEFT REGION VERTICAL LINES. *
96 C      *****
97      DO 3 I=0,KM-1
98      YE=C/(2*KM)+I*C/KM
99      DO 4 J=0,NPX-1
100     K=K+1
101     WRITE(*,*) 'DRAWING VECTOR ',K
102     XE=XWIDTH*(1.0-TAUX)/2.0+J*DX
103     CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
104     NDIR=0
105     VL=20.0/GS
106     WRITE(1,*) '% begin direction 1 of vector ',K
107     CALL CVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
108     & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG,S
109     & ,MNAR,DS,XCMIN,XCMAX,ICLIP)
110     WRITE(1,*) '% end direction 1 of vector ',K
111     NDIR=1
112     VL=20.0/GS-S/GS
113     WRITE(1,*) '% begin direction 2 of vector ',K
114     CALL CVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
115     & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG,S
116     & ,MNAR,DS,XCMIN,XCMAX,ICLIP)
117     WRITE(1,*) '% begin direction 2 of vector ',K
118 4      CONTINUE
119 3      CONTINUE
120 C      *****
121 C      * CENTER REGION HORIZONTAL LINES. *
122 C      *****
123     XE=PI*A
124     DO 5 I=0,KM
125     YZ=-C/(2*KM)+I*C/KM

```

```

126      DO 6 J=0,NPY-1
127      YE=YZ+(C/KM)*(1.0-TAUY)/2.0+J*DY
128      IF ((YE .GT. DY/2.0) .AND. (YE .LT. C-DY/2.0)) THEN
129          K=K+1
130          WRITE(*,*) 'DRAWING VECTOR ',K
131          CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
132          NDIR=0
133          VL=20.0/GS
134          WRITE(1,*) '% begin direction 1 of vector ',K
135          CALL CVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
136          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG,S
137          & ,MNAR,DS,XCMIN,XCMAX,ICLIP)
138          WRITE(1,*) '% end direction 1 of vector ',K
139          NDIR=1
140          VL=20.0/GS-S/GS
141          WRITE(1,*) '% begin direction 2 of vector ',K
142          CALL CVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
143          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG,S
144          & ,MNAR,DS,XCMIN,XCMAX,ICLIP)
145          WRITE(1,*) '% end direction 2 of vector ',K
146      ELSE
147          END IF
148      6  CONTINUE
149      5  CONTINUE
150 C  ****
151 C  * RIGHT REGION VERTICAL LINES. *
152 C  ****
153      DO 7 I=0,KM-1
154      YE=C/(2*KM)+I*C/KM
155      DO 8 J=0,NPX-1
156          K=K+1
157          WRITE(*,*) 'DRAWING VECTOR ',K
158          XE=PI*A+XWIDTH*(1.0-TAUX)/2.0+J*DX
159          CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
160          NDIR=0
161          VL=20.0/GS
162          WRITE(1,*) '% begin direction 1 of vector ',K
163          CALL CVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
164          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG,S
165          & ,MNAR,DS,XCMIN,XCMAX,ICLIP)
166          WRITE(1,*) '% end direction 1 of vector ',K
167          NDIR=1
168          VL=20.0/GS-S/GS
169          WRITE(1,*) '% begin direction 2 of vector ',K
170          CALL CVEC3D(XE,YE,ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
171          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG,S
172          & ,MNAR,DS,XCMIN,XCMAX,ICLIP)
173          WRITE(1,*) '% end direction 2 of vector ',K
174      8  CONTINUE
175      7  CONTINUE
176 C  ****
177 C  * RIGHT REGION HORIZONTAL LINES. *
178 C  ****
179      XE=TP*A
180      DO 9 I=0,KM
181      YZ=-C/(2*KM)+I*C/KM
182      DO 10 J=0,NPY-1
183          YE=YZ+(C/KM)*(1.0-TAUY)/2.0+J*DY
184          IF ((YE .GT. DY/2.0) .AND. (YE .LT. C-DY/2.0)) THEN

```

```

185          K=K+1
186          WRITE(*,*) 'DRAWING VECTOR ',K
187          WRITE(1,*) '% begin direction 1 of vector ',K
188          CALL FNDTRI(XE,YE,MN,X,Y,MT,NT,N2T,ID)
189          NDIR=0
190          VL=20.0/GS
191          CALL CVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
192          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG,S
193          & ,MNAR,DS,XCMIN,XCMAX,ICLIP)
194          WRITE(1,*) '% end direction 1 of vector ',K
195          NDIR=1
196          VL=20.0/GS-S/GS
197          WRITE(1,*) '% begin direction 2 of vector ',K
198          CALL CVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
199          & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG,S
200          & ,MNAR,DS,XCMIN,XCMAX,ICLIP)
201          WRITE(1,*) '% end direction 2 of vector ',K
202          ELSE
203          END IF
204 10      CONTINUE
205 9       CONTINUE
206 99      CONTINUE
207      RETURN
208      END

1      SUBROUTINE CVEC3D(XE,YE, ID,NDIR,MN,MT,MP,NT,N2T,T2N,X,Y,XMIN,XMAX
2      & ,YMIN,YMAX,FX,FY,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,VL,BANG,S
3      & ,MNAR,DS,XCMIN,XCMAX,ICLIP)
4 C      ****
5 C      * THIS SUBROUTINE DRAWS A VECTOR LINE. *
6 C      ****
7 C      * INPUTS: *
8 C      *
9 C      * XE,YE - FIRST COORDINATE OF VECTOR LINE. *
10 C      * MN - MAXIMUM NUMBER OF COORDINATE POINTS. *
11 C      * MT - MAXIMUM NUMBER OF TRIANGLES. *
12 C      * N2T(MT,3) - NODE MATRIX WHERE FIRST INDEX REPRESENTS *
13 C      * EACH TRIANGLE AND SECOND INDEX REPRESENTS *
14 C      * THE NODE NUMBERS. *
15 C      * T2N(MN,MP) - MATRIX OF TRIANGLE TO NODE CONNECTIONS *
16 C      * WHERE THE FIRST INDEX IS THE NODE NUMBER *
17 C      * AND THE SECOND IS THE LOCAL TRIANGLE NUMBER. *
18 C      * X(MN) - COORDINATE VECTORS OF GRID. *
19 C      * Y(MN) *
20 C      * FX(MN) - VECTOR FUNCTION FIELD VALUES AT EACH COORDINATE. *
21 C      * FY(MN) *
22 C      * AX,BX - GRAPHICAL SCALE CONSTANTS. *
23 C      * AY,BY *
24 C      * VL - SPACING BETWEEN ARROWHEADS IN REAL COORDINATES. *
25 C      * BANG - ARROW APEX HALF ANGLE IN DEGREES. *
26 C      * S - ARROW HEAD SIDE LENGTH IN POINT SIZE. *
27 C      * MNAR - MAXIMUM NUMBER OF ARROWS ALLOWED. *
28 C      * DS - STEP SIZE. *
29 C      *
30 C      * OUTPUTS: *
31 C      *
32 C      ****
33      REAL*4 X(MN),Y(MN),FX(MN),FY(MN)
34      INTEGER N2T(MT,3),T2N(MN,MP)

```

```

35      RADIUS=0.5
36 C      ****
37 C      * STORE THE FIRST TRIANGLE NUMBER. *
38 C      ****
39      IFIRST=ID
40 C      ****
41 C      * STORE THE FIRST POINT. *
42 C      ****
43      XFIRST=XE
44      YFIRST=YE
45 C      ****
46 C      * BEGIN THE VECTOR SEGMENTS. *
47 C      ****
48      K=0
49      VSUM=0.0
50      KAR=0
51 999    CONTINUE
52      K=K+1
53 C      WRITE(*,*) 'SEGMENT NUMBER= ',K,' TRIANGLE NUMBER= ',ID
54 C      ****
55 C      * IF THIS IS NOT THE FIRST LINE THEN DETERMINE WHAT TRIANGLE *
56 C      * THE DRAWN LINE CONNECTS TO BY LOOKING AT THE TWO NODES NA AND *
57 C      * NB FORMING THE LAST INTERSECTION. *
58 C      ****
59      IF (K .GT. 1) THEN
60          IF (K .GT. 300) THEN
61              WRITE(*,*) 'MAX STEPS EXCEEDED '
62              WRITE(1,*) '% maximum steps exceeded'
63              GO TO 99
64          ELSE
65          END IF
66 C      ****
67 C      * SET THE BASE LOCATION OF THE VECTOR TO THE HEAD LOCATION OF *
68 C      * THE PREVIOUS VECTOR. *
69 C      ****
70      XA=XF
71      YA=YF
72      IF (ID .EQ. 0) THEN
73 C          ****
74 C          * THERE IS NO NEXT TRIANGLE WHICH MEANS THE VECTOR *
75 C          * TERMINATES ON A BOUNDARY. *
76 C          ****
77      WRITE(*,*) 'TERMINATE ON BOUNDARY TRIANGLE'
78      GO TO 99
79      ELSE IF (ID .EQ. IFIRST) THEN
80          WRITE(*,*) 'BACK TO ORIGINAL TRIANGLE'
81          GO TO 99
82      ELSE
83      END IF
84      ELSE
85 C          ****
86 C          * SET THE BASE LOCATION OF THE VECTOR TO THE INITIAL POINT. *
87 C          ****
88      XA=XE
89      YA=YE
90      END IF
91 C      ****
92 C      * FIND THE VECTOR FUNCTION VALUE AT THE BASE POINT LOCATION *
93 C      * (XA,YA). *

```

```

94 C ****
95 C CALL VECQOF(MN,MT,ID,X,Y,N2T,FX,FY,A,B,E,C,D,F)
96 C ****
97 C * GET THE VECTOR COMPONENTS AT THE POINT (XA,YA). *
98 C ****
99 FYA=A*XA+B*YA+E
100 FXA=C*XA+D*YA+F
101 FM=SQRT(FXA*FXA+FYA*FYA)
102 FXN=FXA/FM
103 FYN=FYA/FM
104 C ****
105 C * COMPUTE THE LOCATION (XF,YF) OF THE HEAD OF THE VECTOR. *
106 C ****
107 IF (NDIR .EQ. 0) THEN
108   XF=XA+DS*FXN
109   YF=YA+DS*FYN
110 ELSE
111   XF=XA-DS*FXN
112   YF=YA-DS*FYN
113 END IF
114 C ****
115 C * IF THIS IS THE FIRST TIME THROUGH THE LOOP THEN CHECK TO SEE *
116 C * IF THE ENTIRE VECTOR SHOULD BE SKIPPED. *
117 C ****
118 IF (K .EQ. 1) THEN
119   IF ((XF .LT. XMIN) .OR. (XF .GT. XMAX) .OR.
120     & (YF .LT. YMIN) .OR. (YF .GT. YMAX)) THEN
121     WRITE(1,*) '% vector skipped'
122     GO TO 104
123   ELSE
124     END IF
125   ELSE
126   END IF
127 C ****
128 C * ADJUST THE HEAD POINT IF IT IS OUTSIDE THE GRAPH BOUNDARY. *
129 C ****
130 IF (XF .LT. XMIN) THEN
131   XF=XMIN
132 ELSE IF (XF .GT. XMAX) THEN
133   XF=XMAX
134 ELSE
135 END IF
136 IF (YF .LT. YMIN) THEN
137   YF=YMIN
138 ELSE IF (YF .GT. YMAX) THEN
139   YF=YMAX
140 ELSE
141 END IF
142 C ****
143 C * CHECK THE BASE AND HEAD POINTS OF THE VECTOR AGAINST THE *
144 C * CLIPPING BOUNDARY. *
145 C ****
146 IF ((XA .GE. XCMIN) .AND. (XA .LE. XCMAX)) THEN
147 C ****
148 C * THE BASE POINT IS WITHIN THE CLIPPING BOUNDARY. *
149 C ****
150 IAFLAG=1
151 ELSE
152 C ****

```

```

153 C      * THE BASE POINT IS OUTSIDE THE CLIPPING BOUNDARY. *
154 C ****
155 C      IAFLAG=2
156 C END IF
157 C IF ((XF .GE. XCMIN) .AND. (XF .LE. XCMAX)) THEN
158 C ****
159 C      * THE HEAD POINT IS WITHIN OR ON THE CLIPPING BOUNDARY. *
160 C ****
161 C      IFFLAG=1
162 C ELSE
163 C ****
164 C      * THE HEAD POINT IS OUTSIDE THE CLIPPING BOUNDARY. *
165 C ****
166 C      IFFLAG=2
167 C END IF
168 C ****
169 C      * GET THE TRIANGLE NUMBER CONTAINING THE HEAD POINT. *
170 C ****
171 C IDOLD=ID
172 C CALL FDSTRI(XF,YF,MN,X,Y,MT,MP,NT,N2T,T2N,IDL0D, ID)
173 C      write(*,*) K,IAFLAG,IFFLAG
174 C      write(1,*) '% K IAFLAG IFFLAG ',K,IAFLAG,IFFLAG
175 C ****
176 C      * HANDLE EACH CASE FOR DRAWING OR NOT DRAWING A VECTOR. *
177 C ****
178 C IF (((IAFLAG .EQ. 1) .AND. (IFFLAG .EQ. 1) .AND. (ICLIP .EQ. 1))
179 & .OR. ((IAFLAG .EQ. 2) .AND. (IFFLAG .EQ. 2)
180 & .AND. (ICLIP .EQ. 2))) THEN
181 C ****
182 C      * THE BASE AND HEAD ARE BOTH VISIBLE. *
183 C ****
184 C IF (K .EQ. 1) THEN
185 C      CALL MPCPNT(XA,YA,RADIUS,XAT,YAT,ZAT)
186 C      CALL TRFPNT(XAT,YAT,ZAT,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
187 C      WRITE(1,100) XAT,YAT,' moveto'
188 C ELSE
189 C      CALL MPCPNT(XF,YF,RADIUS,XFT,YFT,ZFT)
190 C      CALL TRFPNT(XFT,YFT,ZFT,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
191 C      WRITE(1,100) XFT,YFT,' lineto'
192 C END IF
193 C ****
194 C      * DRAW ARROW OR ARROWS IF APPROPRIATE. *
195 C ****
196 C DAR=SQRT((XA-XF)*(XA-XF)+(YA-YF)*(YA-YF))
197 C IF (DAR .GT. 0.0) THEN
198 C      IF (NDIR .EQ. 0) THEN
199 C          CALL CARR3D(MNAR,VL,BANG,S,XA,YA,XF,YF,FXN,FYN,VSUM,KAR
200 &           ,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,IAFLAG,IFFLAG,ICLIP)
201 C      ELSE
202 C          CALL CARR3D(MNAR,VL,BANG,S,XF,YF,XA,YA,FXN,FYN,VSUM,KAR
203 &           ,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,IAFLAG,IFFLAG,ICLIP)
204 C      END IF
205 C ELSE
206 C END IF
207 C ELSE IF ( ((IAFLAG .EQ. 1) .AND. (IFFLAG .EQ. 2) .AND.
208 &             (ICLIP .EQ. 1) .AND. (K .NE. 1)) .OR.
209 &             ((IAFLAG .EQ. 2) .AND. (IFFLAG .EQ. 1) .AND.
210 &             (ICLIP .EQ. 2) .AND. (K .NE. 1)) ) THEN
211 C ****

```

```

212 C      * THE BASE IS VISIBLE AND THE HEAD IS NOT VISIBLE. THIS      *
213 C      * OCCURS AT A CLIPPING BOUNDARY.                                *
214 C      ****
215      CALL MPCPNT(XF,YF,RADIUS,XFT,YFT,ZFT)
216      CALL TRFPNT(XFT,YFT,ZFT,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
217      WRITE(1,101) XFT,YFT,' lineto stroke'
218      WRITE(1,*) '% terminated at clipping boundary'
219      ELSE IF (((IAFLAG .EQ. 2) .AND. (IFFLAG .EQ. 1) .AND.
220      &           (ICLIP .EQ. 1)) .OR. ((IAFLAG .EQ. 1) .AND.
221      &           (IFFLAG .EQ. 2) .AND.(ICLIP .EQ. 2))) THEN
222      CALL MPCPNT(XF,YF,RADIUS,XFT,YFT,ZFT)
223      CALL TRFPNT(XFT,YFT,ZFT,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
224      WRITE(1,*) '% start of visible vector section'
225      WRITE(1,*) ' newpath'
226      WRITE(1,100) XFT,YFT,' moveto'
227      ELSE
228 C      ****
229 C      * BOTH POINTS ARE HIDDEN. JUST ACCUMULATE THE VECTOR LENGTH. *
230 C      ****
231      DAR=SQRT((XA-XF)*(XA-XF)+(YA-YF)*(YA-YF))
232      IF (DAR .GT. 0.0) THEN
233          IF (NDIR .EQ. 0) THEN
234              CALL CARR3D(MNAR,VL,BANG,S,XA,YA,XF,YF,FXN,FYN,VSUM,KAR
235              & ,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,IAFLAG,IFFLAG,ICLIP)
236          ELSE
237              CALL CARR3D(MNAR,VL,BANG,S,XF,YF,XA,YA,FXN,FYN,VSUM,KAR
238              & ,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,IAFLAG,IFFLAG,ICLIP)
239          END IF
240      ELSE
241          END IF
242      END IF
243 C      ****
244 C      * STOP IF VECTOR MAGNITUDE DROPS BELOW TOLERANCE.          *
245 C      ****
246      IF (FM .LT. .001) THEN
247          WRITE(1,*) '% magnitude below tolerance'
248          GO TO 99
249      ELSE
250          END IF
251          GO TO 999
252 C      ****
253 C      * STOP.                                                       *
254 C      ****
255 99      CONTINUE
256      IF ( ((IAFLAG .EQ. 1).OR.(IFFLAG .EQ. 1)) .AND. (ICLIP .EQ. 1))
257      & .OR.
258      & ((IAFLAG .EQ. 2) .OR. (IFFLAG .EQ. 2)) .AND. (ICLIP .EQ. 2)))
259      & THEN
260          WRITE(1,*) 'stroke'
261      ELSE
262          END IF
263 100      FORMAT(2(F6.2,1X),A7)
264 101      FORMAT(2(F6.2,1X),A14)
265 104      CONTINUE
266      ID=IFIRST
267      RETURN
268      END

```

1 SUBROUTINE CARR3D(MNAP,VL,BANG,S,XF,YF,XA,YA,FXN,FYN,VSUM,KAR

```

2      &      ,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0,IAFLAG,IFFLAG,ICLIP)
3 C   ****
4 C   * THIS SUBROUTINE CONTROLS THE DRAWING OF ARROWHEADS ON A LINE. *
5 C   ****
6 C   * INPUTS:          *
7 C   *
8 C   *   MNAR    - MAXIMUM NUMBER OF ARROWS ALLOWED.          *
9 C   *   VL       - SPACING BETWEEN ARROWHEADS IN REAL COORDINATES. *
10 C   *   BANG    - ARROW APEX HALF ANGLE IN DEGREES.          *
11 C   *   S        - ARROW HEAD SIDE LENGTH IN POINT SIZE.       *
12 C   *   XA,YA   - BASE OF ARROW.          *
13 C   *   XF,YF   - TIP OF ARROW.          *
14 C   *   FXN,FYN - NORMALIZED VECTOR FUNCTION VALUES.          *
15 C   *   VSUM    - ACCUMULATED LINE LENGTH.          *
16 C   *   KAR     - NUMBER OF ARROWS DRAWN SO FAR.          *
17 C   *   AX,BX   - GRAPHICAL SCALE CONSTANTS.          *
18 C   *   AY,BY   -          *
19 C   ****
20 D=SQRT((XA-XF)*(XA-XF)+(YA-YF)*(YA-YF))
21 STEP=VSUM+D
22 IF (STEP .GT. VL) THEN
23     KAR=KAR+1
24     IF (KAR .GT. MNAR) THEN
25         GO TO 2
26     ELSE
27     END IF
28 C   ****
29 C   * THE NEXT LINE SEGMENT IS LONGER THAN THE ARROW SPACING.      *
30 C   * SO AN ARROW SHOULD BE DRAWN.          *
31 C   ****
32 RAT=(VL-VSUM)/D
33 XFW=XA+RAT*(XF-XA)
34 YFW=YA+RAT*(YF-YA)
35 C   ****
36 C   * DRAW THE ARROW.          *
37 C   ****
38 IF ( ((IAFLAG .EQ. 1).AND.(IFFLAG .EQ. 1) .AND. (ICLIP .EQ. 1))
39 & .OR.
40 & ((IAFLAG .EQ. 2) .AND. (IFFLAG .EQ. 2) .AND. (ICLIP .EQ. 2)))
41 & THEN
42     CALL CAHD3D(BANG,GS,S,XFW,YFW,FXN,FYN,XD,YD,ZD,C1,C2
43 & ,C3,C4,C5,X0,Y0)
44 ELSE
45 END IF
46 C   ****
47 C   * COMPUTE THE AMOUNT LEFT OVER AFTER THE ARROW IS DRAWN.      *
48 C   ****
49 DLT=SQRT((XF-XFW)*(XF-XFW)+(YF-YFW)*(YF-YFW))
50 IF (DLT .LT. VL) THEN
51 C   ****
52 C   * NO MORE ARROWS ARE DRAWN ON THIS SEGMENT.          *
53 C   ****
54 VSUM=DLT
55 ELSE
56 C   ****
57 C   * MORE THAN ONE ARROW IS DRAWN ON THIS SEGMENT.          *
58 C   ****
59 NAR=DLT/VL
60 DO 1 I=1,NAR

```

```

61      KAR=KAR+1
62      IF (KAR .GT. MNAR) THEN
63          GO TO 2
64      ELSE
65      END IF
66      RAT=(VL-VSUM+I*VL)/D
67      XFW=XA+RAT*(XF-XA)
68      YFW=YA+RAT*(YF-YA)
69      IF ( ((IAFLAG .EQ. 1).AND.(IFFLAG .EQ. 1) .AND. (ICLIP .EQ. 1))
70      & .OR.
71      & ((IAFLAG .EQ. 2) .AND. (IFFLAG .EQ. 2) .AND. (ICLIP .EQ. 2)))
72      & THEN
73          CALL CAHD3D(BANG,GS,S,XFW,YFW,FXN,FYN,XD,YD,ZD,C1,C2
74      & ,C3,C4,C5,X0,Y0)
75      ELSE
76      END IF
77      1    CONTINUE
78      VSUM=SQRT((XF-XFW)*(XF-XFW)+(YF-YFW)*(YF-YFW))
79      END IF
80      ELSE
81 C      ****
82 C      * THE LINE LENGTH PLUS THE ACCUMULATED DISTANCE IS STILL LESS *
83 C      * THAN THE ARROW SPACING. NO ARROW SHOULD BE DRAWN JUST        *
84 C      * ACCUMULATE THE DISTANCE.                                     *
85 C      ****
86      VSUM=VSUM+D
87      END IF
88      2    CONTINUE
89      RETURN
90      END

1      SUBROUTINE CAHD3D(BANG,GS,S,X2,Y2,FXN,FYN,XD,YD,ZD,C1,C2
2      &,C3,C4,C5,X0,Y0)
3 C      ****
4 C      * THIS SUBROUTINE DRAWS A ARROW FROM (X1,Y1) TO (X2,Y2).       *
5 C      ****
6 C      * INPUTS:                                         *
7 C      *                                         *
8 C      *     BANG - ARROW APEX HALF ANGLE IN DEGREES.                 *
9 C      *     S   - ARROWHEAD SIDE LENGTH.                                *
10 C      *    X2,Y2 - TIP OF ARROW.                                    *
11 C      *    FXN,FYN - NORMALIZED VECTOR FUNCTION VALUES.             *
12 C      ****
13 RAD=.17453293E-01
14 X1=X2-FXN
15 Y1=Y2-FYN
16 D=SQRT((X2-X1)*(X2-X1)+(Y2-Y1)*(Y2-Y1))
17 A=1.0-S/(GS+D)
18 X3=X1+A*(X2-X1)
19 Y3=Y1+A*(Y2-Y1)
20 C      ****
21 C      * TRANSLATE THE ORIGIN TO (X2,Y2) AND ROTATE (X3,Y3) PLUS AND   *
22 C      * MINUS BANG DEGREES TO GENERATE THE VERTICES OF THE ARROW.      *
23 C      ****
24 XA=X3-X2
25 YA=Y3-Y2
26 ARG=RAD*BANG
27 CA=COS(ARG)
28 SA=SIN(ARG)

```

```

29      XCA=XA*CA
30      XSA=XA*SA
31      YCA=YA*CA
32      YSA=YA*SA
33      X4=X2+XCA-YSA
34      Y4=Y2+XSA+YCA
35      X5=X2+XCA+YSA
36      Y5=Y2-XSA+YCA
37 C      X6=(X4+X5)/2.0
38 C      Y6=(Y4+Y5)/2.0
39 C      WRITE(1,*) X1,Y1,' moveto'
40 C      WRITE(1,*) X6,Y6,' lineto stroke'
41 C      ****
42 C      * TRANSLATE THE ORIGIN TO (X2,Y2) AND ROTATE (X3,Y3) PLUS AND *
43 C      * MINUS BANG DEGREES TO GENERATE THE VERTICES OF THE ARROW *
44 C      ****
45      RADIUS=0.5
46      CALL MPCPNT(X2,Y2,RADIUS,X2T,Y2T,Z2T)
47      CALL MPCPNT(X4,Y4,RADIUS,X4T,Y4T,Z4T)
48      CALL MPCPNT(X5,Y5,RADIUS,X5T,Y5T,Z5T)
49      CALL TRFPNT(X2T,Y2T,Z2T,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
50      CALL TRFPNT(X4T,Y4T,Z4T,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
51      CALL TRFPNT(X5T,Y5T,Z5T,GS,XD,YD,ZD,C1,C2,C3,C4,C5,X0,Y0)
52      WRITE(1,*) 'gsave newpath'
53      WRITE(1,*) X2T,Y2T,' moveto ',X4T,Y4T,' lineto '
54      &           ,X5T,Y5T,' lineto'
55      WRITE(1,*) 'closepath 0 setgray fill stroke grestore'
56      RETURN
57      END

1      SUBROUTINE MPCPNT(U,V,A,X,Y,Z)
2 C      ****
3 C      * THIS SUBROUTINE MAPS A POINT FROM THE UNFOLDED CIRCULAR *
4 C      * WAVEGUIDE (U,V) SPACE TO THE (X,Y,Z) FOLDED WAVEGUIDE SPACE. *
5 C      ****
6 C      * INPUTS: *
7 C      *
8 C      *   U,V - COORDINATES OF THE UNFOLDED WAVEGUIDE POINT. *
9 C      *   A   - RADIUS OF CIRCULAR WAVEGUIDE (WAVELENGTHS). *
10 C      *
11 C      * OUTPUTS: *
12 C      *
13 C      *   X,Y,Z - COORDINATES OF MAPPED POINT. *
14 C      ****
15      X=A*COS(U/A)
16      Y=A*SIN(U/A)
17      Z=V
18      RETURN
19      END

```

```

1      SUBROUTINE TRFTRI(X,Y,Z,GS,XC,YC,ZC,C1,C2,C3,C4,C5,X0,Y0)
2 C ****
3 C * THIS SUBROUTINE TRANSFORMS EACH POINT OF A TRIANGLE. *
4 C ****
5 C * INPUTS: *
6 C *
7 C * X(3) - COORDINATES OF EACH VERTEX OF A TRIANGLE. *
8 C * Y(3) *
9 C * Z(3) *
10 C * GS - GRAPH SCALE FACTOR. *
11 C * XC,YC,ZC - CENTER OF ROTATION. *
12 C * C1,C2,C3 - ROTATION MATRIX COEFFICIENTS. *
13 C * C4,C5 *
14 C * X0,Y0 - HORIZONTAL AND VERTICAL DEVICE OFFSETS. *
15 C *
16 C * OUTPUT: *
17 C *
18 C * X(3) - TRANSFORMED COORDINATES OF EACH CORNER OF THE *
19 C * Y(3) PROJECTED ONTO THE X-Y PLANE. *
20 C ****
21 REAL*4 X(3),Y(3),Z(3)
22 DO 1 I=1,3
23     CALL SCALE(GS,X(I),Y(I),Z(I))
24     CALL T3LATE(X(I),Y(I),Z(I),XC,YC,ZC)
25     CALL ROTATE(X(I),Y(I),Z(I),C1,C2,C3,C4,C5)
26     CALL T3LATE(X(I),Y(I),Z(I),-XC,-YC,-ZC)
27     CALL T2LATE(X(I),Y(I),X0,Y0)
28 1 CONTINUE
29 RETURN
30 END

1      SUBROUTINE TRFPNT(X,Y,Z,GS,XC,YC,ZC,C1,C2,C3,C4,C5,X0,Y0)
2 C ****
3 C * THIS SUBROUTINE TRANSFORMS A POINT. *
4 C ****
5 C * INPUTS: *
6 C *
7 C * X,Y,Z - COORDINATES OF A POINT. *
8 C * GS - GRAPH SCALE FACTOR. *
9 C * XC,YC,ZC - CENTER OF ROTATION. *
10 C * C1,C2,C3 - ROTATION MATRIX COEFFICIENTS. *
11 C * C4,C5 *
12 C * X0,Y0 - HORIZONTAL AND VERTICAL DEVICE OFFSETS. *
13 C *
14 C * OUTPUT: *
15 C *
16 C * X,Y - TRANSFORMED COORDINATES OF EACH CORNER OF THE *
17 C * PROJECTED ONTO THE X-Y PLANE. *
18 C ****
19 CALL SCALE(GS,X,Y,Z)
20 CALL T3LATE(X,Y,Z,XC,YC,ZC)
21 CALL ROTATE(X,Y,Z,C1,C2,C3,C4,C5)
22 CALL T3LATE(X,Y,Z,-XC,-YC,-ZC)
23 CALL T2LATE(X,Y,X0,Y0)
24 RETURN
25 END

1      SUBROUTINE T3LATE(X,Y,Z,XC,YC,ZC)
2 C ****

```

```

3 C * THIS SUBROUTINE TRANSLATES THE ORIGIN TO THE POINT (XC,YC,ZC). *
4 C ****
5 C X=X-XC
6 C Y=Y-YC
7 C Z=Z-ZC
8 C RETURN
9 C END

1 C SUBROUTINE T2LATE(X,Y,XC,YC)
2 C ****
3 C * THIS SUBROUTINE TRANSLATES THE POINT (X,Y) TO THE POINT      *
4 C * TO THE POINT (XC,YC).                                     *
5 C ****
6 C X=X+XC
7 C Y=Y+YC
8 C RETURN
9 C END

1 C SUBROUTINE SCALE(GS,X,Y,Z)
2 C ****
3 C * THIS SUBROUTINE SCALES A POINT.                         *
4 C ****
5 C X=GS*X
6 C Y=GS*Y
7 C Z=GS*Z
8 C RETURN
9 C END

1 C SUBROUTINE RCOEFF(RP,RT,C1,C2,C3,C4,C5)
2 C ****
3 C * THIS SUBROUTINE COMPUTES THE ROTATION COEFFICIENTS.   *
4 C ****
5 C CP=COS(RP)
6 C SP=SIN(RP)
7 C C1=CP
8 C C2=-SP
9 C CT=COS(RT)
10 C3=CT*SP
11 C4=CT*CP
12 C5=SIN(RT)
13 C RETURN
14 C END

1 C SUBROUTINE ROTATE(X,Y,Z,C1,C2,C3,C4,C5)
2 C ****
3 C * THIS SUBROUTINE ROTATES POINTS IN THREE DIMENSIONS AND TAKES  *
4 C * THE PARALLEL PROJECTION ONTO THE X-Y PLANE.                   *
5 C ****
6 C XR=X
7 C YR=Y
8 C X=C1*XR+C2*YR
9 C Y=C3*XR+C4*YR+C5*Z
10 C RETURN
11 C END

```

```

1      SUBROUTINE VECFCOF(MN,MT, ID ,X,Y,XA,YA,N2T,FX,FY,A,B,E,C,D,F)
2 C ****
3 C * THIS SUBROUTINE RETURNS THE VECTOR FUNCTION COEFFICIENTS FOR *
4 C * THE IDTH TRIANGLE.
5 C ****
6 REAL*4 X(MN),Y(MN),FX(MN),FY(MN)
7 INTEGER N2T(MT,3)
8 X1=X(N2T(ID,1))
9 Y1=Y(N2T(ID,1))
10 X2=X(N2T(ID,2))
11 Y2=Y(N2T(ID,2))
12 X3=X(N2T(ID,3))
13 Y3=Y(N2T(ID,3))
14 FX1=FX(N2T(ID,1))
15 FX2=FX(N2T(ID,2))
16 FX3=FX(N2T(ID,3))
17 FY1=FY(N2T(ID,1))
18 FY2=FY(N2T(ID,2))
19 FY3=FY(N2T(ID,3))
20 DEN=X1*Y2-X1*Y3-X2*Y1+X3*Y1+X2*Y3-X3*Y2
21 A=(FY1*(Y2-Y3)-FY2*(Y1-Y3)+FY3*(Y1-Y2))/DEN
22 B=(-FY1*(X2-X3)+FY2*(X1-X3)-FY3*(X1-X2))/DEN
23 E=(FY1*(X2*Y3-X3*Y2)-FY2*(X1*Y3-X3*Y1)+FY3*(X1*Y2-X2*Y1))/DEN
24 C=(FX1*(Y2-Y3)-FX2*(Y1-Y3)+FX3*(Y1-Y2))/DEN
25 D=(-FX1*(X2-X3)+FX2*(X1-X3)-FX3*(X1-X2))/DEN
26 F=(FX1*(X2*Y3-X3*Y2)-FX2*(X1*Y3-X3*Y1)+FX3*(X1*Y2-X2*Y1))/DEN
27 RETURN
28 END

1      SUBROUTINE FNDTRI(XA,YA,MN,X,Y,MT,NT,N2T, ID )
2 C ****
3 C * THIS SUBROUTINE SORTS THROUGH THE TRIANGLES AND FINDS THE ONE *
4 C * CONTAINING THE POINT XA,YA.
5 C ****
6 REAL*4 X(MN),Y(MN)
7 INTEGER N2T(MT,3)
8 ID=0
9 DO 1 I=1,NT
10     X1=X(N2T(I,1))
11     Y1=Y(N2T(I,1))
12     X2=X(N2T(I,2))
13     Y2=Y(N2T(I,2))
14     X3=X(N2T(I,3))
15     Y3=Y(N2T(I,3))
16     CALL PINTRI(XA,YA,X1,Y1,X2,Y2,X3,Y3,IFLAG)
17     IF (IFLAG .EQ. 1) THEN
18         ID=I
19     ELSE
20     END IF
21 1  CONTINUE
22 C   WRITE(*,*) XA,YA,' IN TRIANGLE ',ID
23 RETURN
24 END

1      SUBROUTINE FDSTRI(XA,YA,MN,X,Y,MT,MP,NT,N2T,T2N,IA, ID )
2 C ****
3 C * THIS SUBROUTINE SORTS THROUGH THE TRIANGLES CONNECTED TO THE *
4 C * THREE NODES OF THE IAITH TRIANGLE TO FIND THE ONE CONTAINING *
5 C * THE POINT XA,YA.

```

```

6 C ****
7 REAL*4 X(MN),Y(MN)
8 INTEGER N2T(MT,3),T2N(MN,MP)
9 ID=0
10 C ****
11 C * CHECK TO SEE IF THE POINT IS IN THE IATH TRIANGLE. *
12 C ****
13 DO 2 I=1,3
14     N=N2T(IA,I)
15 C     WRITE(*,*) 'NODE ',N,' CONNECTED TO TRIANGLE ',IA
16     DO 3 J=1,MP
17         IF (T2N(N,J) .NE. 0) THEN
18             WRITE(*,*) 'TRIANGLE NUMBER ',N,J,T2N(N,J)
19             X1=X(N2T(T2N(N,J),1))
20             Y1=Y(N2T(T2N(N,J),1))
21             X2=X(N2T(T2N(N,J),2))
22             Y2=Y(N2T(T2N(N,J),2))
23             X3=X(N2T(T2N(N,J),3))
24             Y3=Y(N2T(T2N(N,J),3))
25             CALL PINTRI(XA,YA,X1,Y1,X2,Y2,X3,Y3,IFLAG)
26             IF (IFLAG .EQ. 1) THEN
27                 ID=T2N(N,J)
28                 GO TO 99
29             ELSE
30                 END IF
31             ELSE
32                 END IF
33     3 CONTINUE
34     2 CONTINUE
35     99 CONTINUE
36     RETURN
37 END

1 SUBROUTINE PINTRI(X,Y,X1,Y1,X2,Y2,X3,Y3,IFLAG)
2 ****
3 C * THIS SUBROUTINE DETERMINES WHETHER THE POINT (X,Y) LIES INSIDE *
4 C * THE TRIANGLE DETERMINED BY THE POINTS (X1,Y1), (X2,Y2) AND *
5 C * (X3,Y3). *
6 C ****
7 IFLAG=0
8 C ****
9 C * CHECK TO SEE IF THE POINT IS IN THE SAME HALF PLANE AS POINT *
10 C * (X1,Y1). *
11 C ****
12 CALL HPLANE(X1,Y1,X2,Y2,X3,Y3,I)
13 CALL HPLANE(X,Y,X2,Y2,X3,Y3,J)
14 IF ((I .EQ. J) .OR. (J .EQ. 0)) THEN
15 C ****
16 C * CHECK TO SEE IF THE POINT (X,Y) IS IN THE SAME HALF PLANE AS *
17 C * POINT (X2,Y2). *
18 C ****
19 CALL HPLANE(X2,Y2,X3,Y3,X1,Y1,I)
20 CALL HPLANE(X,Y,X3,Y3,X1,Y1,J)
21 IF ((I .EQ. J) .OR. (J .EQ. 0)) THEN
22 C ****
23 C * CHECK TO SEE IF THE POINT (X,Y) IS IN THE SAME HALF PLANE *
24 C * AS POINT (X3,Y3). *
25 C ****
26 CALL HPLANE(X3,Y3,X1,Y1,X2,Y2,I)

```

```

27      CALL HPLANE(X,Y,X1,Y1,X2,Y2,J)
28      IF ((I .EQ. J) .OR. (J .EQ. 0)) THEN
29          IFLAG=1
30      ELSE
31          END IF
32      ELSE
33          END IF
34      ELSE
35      END IF
36      RETURN
37  END

1  SUBROUTINE HPLANE(X,Y,X1,Y1,X2,Y2,I)
2 C ****
3 C * THIS SUBROUTINE DETERMINES WHETHER THE POINT (X,Y) LIES ON *
4 C * ABOVE OR BELOW THE LINE DETERMINED BY THE POINTS (X1,Y1) AND *
5 C * (X2,Y2). *
6 C ****
7  REAL*8 D
8  D=(X-X1)*(Y2-Y1)-(X2-X1)*(Y-Y1)
9  IF (D .LT. 0.0) THEN
10     I=-1
11  ELSE IF (D .GT. 0.0) THEN
12     I=1
13  ELSE
14     I=0
15  END IF
16  RETURN
17 END

1  SUBROUTINE CONTUR(X1,Y1,X2,Y2,X3,Y3,Z1,Z2,Z3,CA,CB,NC,LC)
2 C ****
3 C * THIS SUBROUTINE DRAWS CONTOURS ON A SINGLE TRIANGLE. *
4 C ****
5 C * INPUTS: *
6 C *
7 C * (X1,Y1), (X2,Y2), (X3,Y3) - TRIANGLE VERTICES. *
8 C * Z1,Z2,Z3 - TRIANGLE VERTEX FUNCTION VALUES. *
9 C ****
10 REAL CA(NC),CB(NC)
11 INTEGER LC(NC)
12 C ****
13 C * LOOP THROUGH EACH CONTOUR PAIR. *
14 C ****
15 DO 1 I=1,NC
16     CMIN=CA(I)
17     CMAX=CB(I)
18 C ****
19 C * 3 POINTS WITHIN THE CONTOUR RANGE. *
20 C ****
21 IF ((Z1 .GE. CMIN) .AND. (Z1 .LE. CMAX) .AND.
22 & (Z2 .GE. CMIN) .AND. (Z2 .LE. CMAX) .AND.
23 & (Z3 .GE. CMIN) .AND. (Z3 .LE. CMAX)) THEN
24     CALL SETCOL(LC(I))
25     CALL FILTRI(X1,Y1,X2,Y2,X3,Y3)
26 C ****
27 C * VERTEX 1 AND 2 ARE IN RANGE AND VERTEX 3 IS OUT OF RANGE. *
28 C ****
29 ELSE IF (((Z3 .LT. CMIN) .OR. (Z3 .GT. CMAX)).AND.

```

```

30      &          (Z1 .GE. CMIN) .AND. (Z1 .LE. CMAX) .AND.
31      &          (Z2 .GE. CMIN) .AND. (Z2 .LE. CMAX)) THEN
32          IF (Z3 .LT. CMIN) THEN
33              CV=CMIN
34          ELSE
35              CV=CMAX
36          END IF
37          CALL CROSS(CV,X3,Y3,Z3,X1,Y1,Z1,XL,YL)
38          CALL CROSS(CV,X2,Y2,Z2,X3,Y3,Z3,XQ,YQ)
39          CALL SETCOL(LC(I))
40          CALL FILQUD(XL,YL,X1,Y1,X2,Y2,XQ,YQ)

41 C ****
42 C * VERTEX 2 AND 3 ARE IN RANGE AND VERTEX 1 IS OUT OF RANGE. *
43 C ****
44     & ELSE IF (((Z1 .LT. CMIN) .OR. (Z1 .GT. CMAX)).AND.
45     &          (Z2 .GE. CMIN) .AND. (Z2 .LE. CMAX) .AND.
46     &          (Z3 .GE. CMIN) .AND. (Z3 .LE. CMAX)) THEN
47         IF (Z1 .LT. CMIN) THEN
48             CV=CMIN
49         ELSE
50             CV=CMAX
51         END IF
52         CALL CROSS(CV,X1,Y1,Z1,X2,Y2,Z2,XL,YL)
53         CALL CROSS(CV,X3,Y3,Z3,X1,Y1,Z1,XQ,YQ)
54         CALL SETCOL(LC(I))
55         CALL FILQUD(XL,YL,X2,Y2,X3,Y3,XQ,YQ)

56 C ****
57 C * VERTEX 3 AND 1 ARE IN RANGE AND VERTEX 2 IS OUT OF RANGE. *
58 C ****
59     & ELSE IF (((Z2 .LT. CMIN) .OR. (Z2 .GT. CMAX)).AND.
60     &          (Z1 .GE. CMIN) .AND. (Z1 .LE. CMAX) .AND.
61     &          (Z3 .GE. CMIN) .AND. (Z3 .LE. CMAX)) THEN
62         IF (Z2 .LT. CMIN) THEN
63             CV=CMIN
64         ELSE
65             CV=CMAX
66         END IF
67         CALL CROSS(CV,X2,Y2,Z2,X3,Y3,Z3,XL,YL)
68         CALL CROSS(CV,X1,Y1,Z1,X2,Y2,Z2,XQ,YQ)
69         CALL SETCOL(LC(I))
70         CALL FILQUD(XL,YL,X3,Y3,X1,Y1,XQ,YQ)

71 C ****
72 C * VERTEX 1 IS IN RANGE AND VERTEX 2 AND 3 ARE BOTH EITHER *
73 C * ABOVE OR BELOW THE RANGE. *
74 C ****
75     & ELSE IF ((Z1 .GE. CMIN) .AND. (Z1 .LE. CMAX) .AND.
76     &          (((Z2 .LT. CMIN) .AND. (Z3 .LT. CMIN)) .OR.
77     &          ((Z2 .GT. CMAX) .AND. (Z3 .GT. CMAX)))) THEN
78         IF ((Z2 .LT. CMIN) .AND. (Z3 .LT. CMIN)) THEN
79             CV=CMIN
80         ELSE
81             CV=CMAX
82         END IF
83         CALL CROSS(CV,X1,Y1,Z1,X2,Y2,Z2,XL,YL)
84         CALL CROSS(CV,X3,Y3,Z3,X1,Y1,Z1,XQ,YQ)
85         CALL SETCOL(LC(I))
86         CALL FILTRI(X1,Y1,XL,YL,XQ,YQ)

87 C ****
88 C * VERTEX 2 IS IN RANGE AND VERTEX 3 AND 1 ARE BOTH EITHER *

```

```

89 C * ABOVE OR BELOW THE RANGE. *
90 C ****
91 & ELSE IF ((Z2 .GE. CMIN) .AND. (Z2 .LE. CMAX) .AND.
92 &     (((Z3 .LT. CMIN) .AND. (Z1 .LT. CMIN)) .OR.
93 &     ((Z3 .GT. CMAX) .AND. (Z1 .GT. CMAX)))) THEN
94     IF ((Z3 .LT. CMIN) .AND. (Z1 .LT. CMIN)) THEN
95         CV=CMIN
96     ELSE
97         CV=CMAX
98     END IF
99     CALL CROSS(CV,X2,Y2,Z2,X3,Y3,Z3,XL,YL)
100    CALL CROSS(CV,X1,Y1,Z1,X2,Y2,Z2,XQ,YQ)
101    CALL SETCOL(LC(I))
102    CALL FILTRI(X2,Y2,XL,YL,XQ,YQ)
103 C ****
104 C * VERTEX 3 IS IN RANGE AND VERTEX 1 AND 2 ARE BOTH EITHER *
105 C * ABOVE OR BELOW THE RANGE. *
106 C ****
107 & ELSE IF ((Z3 .GE. CMIN) .AND. (Z3 .LE. CMAX) .AND.
108 &     (((Z1 .LT. CMIN) .AND. (Z2 .LT. CMIN)) .OR.
109 &     ((Z1 .GT. CMAX) .AND. (Z2 .GT. CMAX)))) THEN
110     IF ((Z1 .LT. CMIN) .AND. (Z2 .LT. CMIN)) THEN
111         CV=CMIN
112     ELSE
113         CV=CMAX
114     END IF
115     CALL CROSS(CV,X3,Y3,Z3,X1,Y1,Z1,XL,YL)
116     CALL CROSS(CV,X2,Y2,Z2,X3,Y3,Z3,XQ,YQ)
117     CALL SETCOL(LC(I))
118     CALL FILTRI(X3,Y3,XL,YL,XQ,YQ)
119 C ****
120 C * VERTEX 1 IS IN RANGE AND VERTEX 2 IS BELOW AND VERTEX 3   *
121 C * IS ABOVE. *
122 C ****
123 & ELSE IF ((Z1 .GE. CMIN) .AND. (Z1 .LE. CMAX) .AND.
124 &     (Z2 .LT. CMIN) .AND. (Z3 .GT. CMAX)) THEN
125     CALL CROSS(CMIN,X1,Y1,Z1,X2,Y2,Z2,XL,YL)
126     CALL CROSS(CMIN,X2,Y2,Z2,X3,Y3,Z3,XQ,YQ)
127     CALL CROSS(CMAX,X2,Y2,Z2,X3,Y3,Z3,XR,YR)
128     CALL CROSS(CMAX,X3,Y3,Z3,X1,Y1,Z1,XS,YS)
129     CALL SETCOL(LC(I))
130     CALL FILPNT(X1,Y1,XL,YL,XQ,YQ,XR,YR,XS,YS)
131 C ****
132 C * VERTEX 1 IS IN RANGE AND VERTEX 2 IS ABOVE AND VERTEX 3 IS   *
133 C * BELOW RANGE. *
134 C ****
135 & ELSE IF ((Z1 .GE. CMIN) .AND. (Z1 .LE. CMAX) .AND.
136 &     (Z2 .GT. CMAX) .AND. (Z3 .LT. CMIN)) THEN
137     CALL CROSS(CMAX,X1,Y1,Z1,X2,Y2,Z2,XL,YL)
138     CALL CROSS(CMAX,X2,Y2,Z2,X3,Y3,Z3,XQ,YQ)
139     CALL CROSS(CMIN,X2,Y2,Z2,X3,Y3,Z3,XR,YR)
140     CALL CROSS(CMIN,X3,Y3,Z3,X1,Y1,Z1,XS,YS)
141     CALL SETCOL(LC(I))
142     CALL FILPNT(X1,Y1,XL,YL,XQ,YQ,XR,YR,XS,YS)
143 C ****
144 C * VERTEX 2 IS IN RANGE AND VERTEX 3 IS BELOW AND VERTEX 1   *
145 C * IS ABOVE. *
146 C ****
147 ELSE IF ((Z2 .GE. CMIN) .AND. (Z2 .LE. CMAX) .AND.

```

```

148      &          (Z3 .LT. CMIN) .AND. (Z1 .GT. CMAX)) THEN
149      CALL CROSS(CMIN,X2,Y2,Z2,X3,Y3,Z3,XL,YL)
150      CALL CROSS(CMIN,X3,Y3,Z3,X1,Y1,Z1,XQ,YQ)
151      CALL CROSS(CMAX,X3,Y3,Z3,X1,Y1,Z1,XR,YR)
152      CALL CROSS(CMAX,X1,Y1,Z1,X2,Y2,Z2,XS,YS)
153      CALL SETCOL(LC(I))
154      CALL FILPNT(X2,Y2,XL,YL,XQ,YQ,XR,YR,XS,YS)
155 C ****
156 C * VERTEX 2 IS IN RANGE AND VERTEX 3 IS ABOVE AND VERTEX 1 IS *
157 C * BELOW RANGE.                                                 *
158 C ****
159 ELSE IF ((Z2 .GE. CMIN) .AND. (Z2 .LE. CMAX) .AND.
160      &          (Z3 .GT. CMAX) .AND. (Z1 .LT. CMIN)) THEN
161      CALL CROSS(CMAX,X2,Y2,Z2,X3,Y3,Z3,XL,YL)
162      CALL CROSS(CMAX,X3,Y3,Z3,X1,Y1,Z1,XQ,YQ)
163      CALL CROSS(CMIN,X3,Y3,Z3,X1,Y1,Z1,XR,YR)
164      CALL CROSS(CMIN,X1,Y1,Z1,X2,Y2,Z2,XS,YS)
165      CALL SETCOL(LC(I))
166      CALL FILPNT(X2,Y2,XL,YL,XQ,YQ,XR,YR,XS,YS)
167 C ****
168 C * VERTEX 3 IS IN RANGE AND VERTEX 1 IS BELOW AND VERTEX 2 *
169 C * IS ABOVE.                                                 *
170 C ****
171 ELSE IF ((Z3 .GE. CMIN) .AND. (Z3 .LE. CMAX) .AND.
172      &          (Z1 .LT. CMIN) .AND. (Z2 .GT. CMAX)) THEN
173      CALL CROSS(CMIN,X3,Y3,Z3,X1,Y1,Z1,XL,YL)
174      CALL CROSS(CMIN,X1,Y1,Z1,X2,Y2,Z2,XQ,YQ)
175      CALL CROSS(CMAX,X1,Y1,Z1,X2,Y2,Z2,XR,YR)
176      CALL CROSS(CMAX,X2,Y2,Z2,X3,Y3,Z3,XS,YS)
177      CALL SETCOL(LC(I))
178      CALL FILPNT(X3,Y3,XL,YL,XQ,YQ,XR,YR,XS,YS)
179 C ****
180 C * VERTEX 3 IS IN RANGE AND VERTEX 1 IS ABOVE AND VERTEX 2 IS *
181 C * BELOW RANGE.                                                 *
182 C ****
183 ELSE IF ((Z3 .GE. CMIN) .AND. (Z3 .LE. CMAX) .AND.
184      &          (Z1 .GT. CMAX) .AND. (Z2 .LT. CMIN)) THEN
185      CALL CROSS(CMAX,X3,Y3,Z3,X1,Y1,Z1,XL,YL)
186      CALL CROSS(CMAX,X1,Y1,Z1,X2,Y2,Z2,XQ,YQ)
187      CALL CROSS(CMIN,X1,Y1,Z1,X2,Y2,Z2,XR,YR)
188      CALL CROSS(CMIN,X2,Y2,Z2,X3,Y3,Z3,XS,YS)
189      CALL SETCOL(LC(I))
190      CALL FILPNT(X3,Y3,XL,YL,XQ,YQ,XR,YR,XS,YS)
191 C ****
192 C * VERTEX 1 IS BELOW OR ABOVE RANGE AND VERTEX 2 AND 3 ARE BOTH *
193 C * ABOVE OR BELOW RANGE.                                         *
194 C ****
195 ELSE IF (((Z1 .LT. CMIN) .AND. (Z2 .GT. CMAX)
196      &          .AND. (Z3 .GT. CMAX)) .OR.
197      &          ((Z1 .GT. CMAX) .AND. (Z2 .LT. CMIN)
198      &          .AND. (Z3 .LT. CMIN))) THEN
199      CALL CROSS(CMIN,X1,Y1,Z1,X2,Y2,Z2,XL,YL)
200      CALL CROSS(CMAX,X1,Y1,Z1,X2,Y2,Z2,XQ,YQ)
201      CALL CROSS(CMAX,X3,Y3,Z3,X1,Y1,Z1,XR,YR)
202      CALL CROSS(CMIN,X3,Y3,Z3,X1,Y1,Z1,XS,YS)
203      CALL SETCOL(LC(I))
204      CALL FILQUD(XL,YL,XQ,YQ,XR,YR,XS,YS)
205 C ****
206 C * VERTEX 2 IS BELOW OR ABOVE RANGE AND VERTEX 3 AND 1 ARE BOTH *

```

```

207 C      * ABOVE OR BELOW RANGE. *
208 C ****
209 ELSE IF (((Z2 .LT. CMIN) .AND. (Z3 .GT. CMAX)
210 &           .AND. (Z1 .GT. CMAX)) .OR.
211 &           ((Z2 .GT. CMAX) .AND. (Z3 .LT. CMIN)
212 &           .AND. (Z1 .LT. CMIN))) THEN
213     CALL CROSS(CMIN,X2,Y2,Z2,X3,Y3,Z3,XL,YL)
214     CALL CROSS(CMAX,X2,Y2,Z2,X3,Y3,Z3,XQ,YQ)
215     CALL CROSS(CMAX,X1,Y1,Z1,X2,Y2,Z2,XR,YR)
216     CALL CROSS(CMIN,X1,Y1,Z1,X2,Y2,Z2,XS,YS)
217     CALL SETCOL(LC(I))
218     CALL FILQUAD(XL,YL,XQ,YQ,XR,YR,XS,YS)
219 C ****
220 C      * VERTEX 3 IS BELOW OR ABOVE RANGE AND VERTEX 1 AND 2 ARE BOTH *
221 C      * ABOVE OR BELOW RANGE. *
222 C ****
223 ELSE IF (((Z3 .LT. CMIN) .AND. (Z1 .GT. CMAX)
224 &           .AND. (Z2 .GT. CMAX)) .OR.
225 &           ((Z3 .GT. CMAX) .AND. (Z1 .LT. CMIN)
226 &           .AND. (Z2 .LT. CMIN))) THEN
227     CALL CROSS(CMIN,X3,Y3,Z3,X1,Y1,Z1,XL,YL)
228     CALL CROSS(CMAX,X3,Y3,Z3,X1,Y1,Z1,XQ,YQ)
229     CALL CROSS(CMAX,X2,Y2,Z2,X3,Y3,Z3,XR,YR)
230     CALL CROSS(CMIN,X2,Y2,Z2,X3,Y3,Z3,XS,YS)
231     CALL SETCOL(LC(I))
232     CALL FILQUAD(XL,YL,XQ,YQ,XR,YR,XS,YS)
233 ELSE
234   END IF
235 1  CONTINUE
236 RETURN
237 END

1 SUBROUTINE CROSS(CV,XA,YA,ZA,XB,YB,ZB,XI,YI)
2 C ****
3 C      * THIS SUBROUTINE COMPUTES THE INTERSECTION OF A CONTOUR LINE   *
4 C      * WITH A TRIANGLE BOUNDARY LINE. *
5 C ****
6 C      * INPUTS: *
7 C      *
8 C      * CV    - CONTOUR VALUE. *
9 C      * XA,YA - COORDINATES OF END POINTS. *
10 C      * XB,YB *
11 C      * ZA,ZB - FUNCTION VALUES AT THE END POINTS. *
12 C      *
13 C      * OUTPUTS: *
14 C      *
15 C      * XI,YI - INTERCEPT COORDINATES. *
16 C ****
17 C      WA=CV-ZA
18 C      WB=CV-ZB
19 C      DF=(ZB-ZA)
20 C      XI=(WA*XB-WB*XA)/DF
21 C      YI=(WA*YB-WB*YA)/DF
22 C      RETURN
23 C      END

1 SUBROUTINE LEGEND(US,VS,NC,NL,LC,LABEL,SL,SH,DS,ITYPE)
2 C ****
3 C      * THIS SURROUTINE DRAWS THE LEGEND. *

```

```

4 C ****
5 C * DRAW THE COLOR KEY. *
6 C *
7 C * (US,VS) - LOWER LEFT CORNER OF LEGEND. *
8 C * NC - NUMBER OF COLORS. *
9 C * NL - NUMBER OF LABELS. *
10 C * LABEL(NC) - LABEL STRINGS. *
11 C * LC(K) - LABEL RECTANGLE COLOR CODE. *
12 C * SL - SIDE LENGTH OF EACH RECTANGLE. *
13 C * SH - SIDE HEIGHT OF EACH RECTANGLE. *
14 C * DS - SPACING BETWEEN ADJACENT RECTANGLES. *
15 C ****
16 CHARACTER*30 LABEL(NL)
17 CHARACTER*37 PSC
18 INTEGER LC(NC)
19 PSC=') stringwidth pop 2 div neg 0 rmoveto'
20 C ****
21 C * DEFINE THE SQUARES. *
22 C ****
23 IF (ITYPE .EQ. 1) THEN
24 C ****
25 C * DRAW A VERTICAL BLOCK TYPE LEGEND. *
26 C ****
27 DO 1 I=1,NC
28 U=US
29 V=VS+(I-1)*(SH+DS)
30 C ****
31 C * FILL EACH RECTANGLE. *
32 C ****
33 WRITE(1,*) 'gsave'
34 WRITE(1,*) U,V,' moveto'
35 WRITE(1,*) U+SL,V,' lineto'
36 WRITE(1,*) U+SL,V+SH,' lineto'
37 WRITE(1,*) U+SL,V+SH,' lineto'
38 WRITE(1,*) U,V+SH,' lineto closepath'
39 CALL SETCOL(LC(I))
40 WRITE(1,*) 'fill stroke'
41 WRITE(1,*) 'grestore'
42 C ****
43 C * DRAW THE PERIMETER AROUND EACH RECTANGLE. *
44 C ****
45 WRITE(1,*) 'gsave 0 setgray 0.5 setlinewidth'
46 WRITE(1,*) U,V,' moveto'
47 WRITE(1,*) U+SL,V,' lineto'
48 WRITE(1,*) U+SL,V+SH,' lineto'
49 WRITE(1,*) U+SL,V+SH,' lineto'
50 WRITE(1,*) U,V+SH,' lineto closepath stroke grestore'
51 C ****
52 C * DRAW THE RECTANGLE LABEL. *
53 C ****
54 WRITE(1,*) '0 setgray'
55 WRITE(1,*) '/Times-Roman findfont 10 scalefont setfont'
56 WRITE(1,*) U+SL+2.0,V-3.0,' moveto'
57 CALL CHRLIM(LABEL(I),MINCHR,MAXCHR)
58 WRITE(1,*) '('//LABEL(I)(MINCHR:MAXCHR)//') show'
59 IF (I .EQ. N) THEN
60 WRITE(1,*) '0 setgray'
61 WRITE(1,*) '/Times-Roman findfont 10 scalefont setfont'
62 WRITE(1,*) U+SL+2.0,V+SH-3.0,' moveto'

```

```

63      CALL CHRLIM(LABEL(I+1),MINCHR,MAXCHR)
64      WRITE(1,*)'(''//LABEL(I+1)(MINCHR:MAXCHR)//') show'
65      ELSE
66      END IF
67 1    CONTINUE
68      ELSE
69 C   ****
70 C   * DRAW A HORIZONTAL BLOCK TYPE LEGEND. *
71 C   ****
72      DO 2 I=1,NC
73      U=US+(I-1)*SL
74      V=VS
75 C   ****
76 C   * FILL EACH RECTANGLE. *
77 C   ****
78      WRITE(1,*)'gsave'
79      WRITE(1,*)'U,V,'moveto'
80      WRITE(1,*)'U+SL,V,'lineto'
81      WRITE(1,*)'U+SL,V+SH,'lineto'
82      WRITE(1,*)'U+SL,V+SH,'lineto'
83      WRITE(1,*)'U,V+SH,'lineto closepath'
84      CALL SETCOL(LC(I))
85      WRITE(1,*)'fill stroke'
86      WRITE(1,*)'grestore'
87 C   ****
88 C   * DRAW THE PERIMETER AROUND EACH RECTANGLE. *
89 C   ****
90      WRITE(1,*)'gsave 0 setgray 0.5 setlinewidth'
91      WRITE(1,*)'U,V,'moveto'
92      WRITE(1,*)'U+SL,V,'lineto'
93      WRITE(1,*)'U+SL,V+SH,'lineto'
94      WRITE(1,*)'U+SL,V+SH,'lineto'
95      WRITE(1,*)'U,V+SH,'lineto closepath stroke grestore'
96 C   ****
97 C   * DRAW THE RECTANGLE LABEL. *
98 C   ****
99      WRITE(1,*)'0 setgray'
100     WRITE(1,*)'/Times-Roman findfont 10 scalefont setfont'
101     WRITE(1,*)'U,V-.9*SH,'moveto'
102     CALL CHRLIM(LABEL(I),MINCHR,MAXCHR)
103     WRITE(1,*)'(''//LABEL(I)(MINCHR:MAXCHR)//PSC
104     WRITE(1,*)'(''//LABEL(I)(MINCHR:MAXCHR)//') show'
105     IF (I .EQ. NC) THEN
106         WRITE(1,*)'0 setgray'
107         WRITE(1,*)'/Times-Roman findfont 10 scalefont setfont'
108         WRITE(1,*)'U+SL,V-.9*SH,'moveto'
109         CALL CHRLIM(LABEL(I+1),MINCHR,MAXCHR)
110         WRITE(1,*)'(''//LABEL(I+1)(MINCHR:MAXCHR)//PSC
111         WRITE(1,*)'(''//LABEL(I+1)(MINCHR:MAXCHR)//') show'
112     ELSE
113     END IF
114 2    CONTINUE
115     END IF
116     RETURN
117     END

1      SUBROUTINE CHRLIM(CS,MINCHR,MAXCHR)
2 C   ****
3 C   * THIS SUBROUTINE DETERMINES THE MINIMUM AND MAXIMUM NONBLANK  *

```

```

4 C * CHARACTER IN A CHARACTER STRING. *
5 C ****
6 CHARACTER*(*) CS
7 MAX=LEN(CS)
8 MINCHR=1
9 MAXCHR=MAX
10 DO 1 I=1,MAX
11   IF (CS(I:I) .NE. ' ') THEN
12     MINCHR=I
13     GO TO 2
14   ELSE
15     END IF
16   1 CONTINUE
17   2 CONTINUE
18   DO 3 I=MAX,1,-1
19     IF (CS(I:I) .NE. ' ') THEN
20       MAXCHR=I
21       GO TO 4
22     ELSE
23     END IF
24   3 CONTINUE
25   4 CONTINUE
26   RETURN
27 END

1      SUBROUTINE SETCOL(I)
2 ****
3 C * THIS SUBROUTINE SETS THE DRAWING COLOR. *
4 C ****
5 IF (I .EQ. 0) THEN
6   WRITE(1,*) 'black'
7 ELSE IF (I .EQ. 1) THEN
8   WRITE(1,*) 'white'
9 ELSE IF (I .EQ. 2) THEN
10  WRITE(1,*) 'gray-lt'
11 ELSE IF (I .EQ. 3) THEN
12  WRITE(1,*) 'gray-lt-med'
13 ELSE IF (I .EQ. 4) THEN
14  WRITE(1,*) 'gray'
15 ELSE IF (I .EQ. 5) THEN
16  WRITE(1,*) 'gray-dk-med'
17 ELSE IF (I .EQ. 6) THEN
18  WRITE(1,*) 'gray-dk'
19 ELSE IF (I .EQ. 7) THEN
20  WRITE(1,*) 'red'
21 ELSE IF (I .EQ. 8) THEN
22  WRITE(1,*) 'magenta'
23 ELSE IF (I .EQ. 9) THEN
24  WRITE(1,*) 'green'
25 ELSE IF (I .EQ. 10) THEN
26  WRITE(1,*) 'blue'
27 ELSE IF (I .EQ. 11) THEN
28  WRITE(1,*) 'cyan'
29 ELSE IF (I .EQ. 12) THEN
30  WRITE(1,*) 'yellow'
31 ELSE IF (I .EQ. 13) THEN
32  WRITE(1,*) 'orange'
33 ELSE IF (I .EQ. 14) THEN
34  WRITE(1,*) 'brown'

```

```

35      ELSE IF (I .EQ. 15) THEN
36          WRITE(1,*) 'kakhi'
37      ELSE IF (I .EQ. 16) THEN
38          WRITE(1,*) 'blue-lt'
39      ELSE IF (I .EQ. 17) THEN
40          WRITE(1,*) 'green-lt'
41      ELSE IF (I .EQ. 18) THEN
42          WRITE(1,*) 'green-blue'
43      ELSE IF (I .EQ. 19) THEN
44          WRITE(1,*) 'purple'
45      ELSE
46          WRITE(1,*) 'white'
47      END IF
48      RETURN
49  END

1      SUBROUTINE FILTRI(X1,Y1,X2,Y2,X3,Y3)
2  C **** THIS SUBROUTINE FILLS A TRIANGLE WITH THE CURRENT COLOR. ****
3  C ****
4  C
5      WRITE(1,1) X1,Y1,X2,Y2,X3,Y3,' filtri'
6  1      FORMAT(6(F6.2,1X),A7)
7      RETURN
8  END

1      SUBROUTINE FILQUD(X1,Y1,X2,Y2,X3,Y3,X4,Y4)
2  C **** THIS SUBROUTINE FILLS A QUADRILATERAL WITH THE CURRENT COLOR. ****
3  C ****
4  C
5      WRITE(1,1) X1,Y1,X2,Y2,X3,Y3,X4,Y4,' filqud'
6  1      FORMAT(8(F6.2,1X),A7)
7      RETURN
8  END

1      SUBROUTINE FILPNT(X1,Y1,X2,Y2,X3,Y3,X4,Y4,X5,Y5)
2  C **** THIS SUBROUTINE FILLS A PENTAGON WITH THE CURRENT COLOR. ****
3  C ****
4  C
5      WRITE(1,1) X1,Y1,X2,Y2,X3,Y3,X4,Y4,X5,Y5,' filpnt'
6  1      FORMAT(10(F6.2,1X),A7)
7      RETURN
8  END

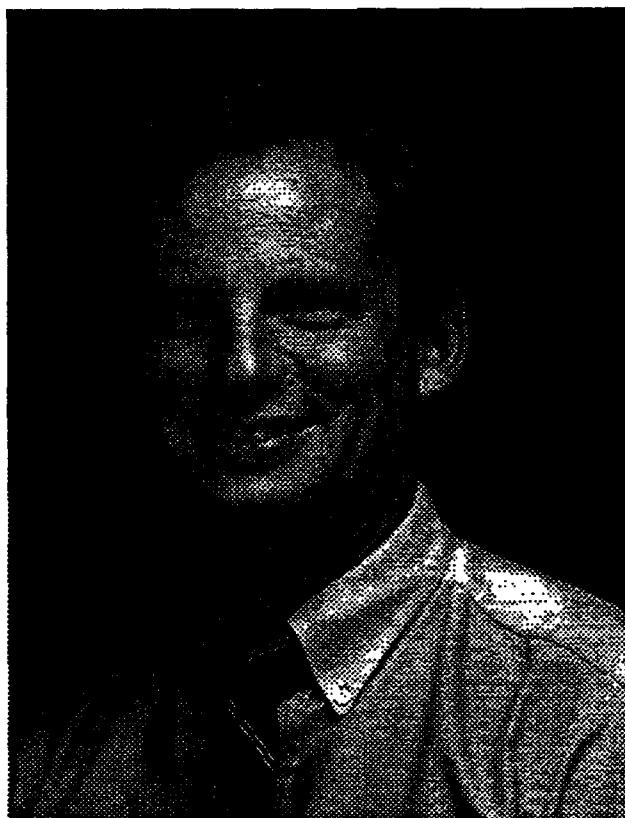
1      FUNCTION TRANS(A,B,S)
2      TRANS=A*S+B
3      RETURN
4  END

```

## APPENDIX B REFERENCES

- [B1] T. M. R. Ellis, "*A structured approach to Fortran 77 programming*," Addison Wesley, 1982.
- [B2] Adobe Systems Inc., "*Postscript language reference manual*," Addison Wesley, 1986.
- [B3] Adobe Systems Inc., "*Postscript language tutorial and cookbook*," Addison Wesley, 1986.
- [B4] G. C. Reid, "*Postscript language program design*," Addison Wesley, 1988.

#### ABOUT THE AUTHOR



Timothy John Peters was born in Pontiac, Michigan on January 25, 1959. He received the B.A. degree in Accounting and the B.S. degree in Electrical Engineering from Michigan State University, East Lansing in 1983 and the M.S. and Ph.D. degrees in Electrical Engineering from The University of Michigan, Ann Arbor in 1984 and 1988 respectively.

He is currently with The Aerospace Corporation, Antennas and Propagation Department in El Segundo, California. His primary research activity is the development of numerical methods for the analysis of antenna and scattering problems.